Incremental Anomaly Detection with Guarantee in the Internet of Medical Things

Xiayan Ji
xjiae@seas.upenn.edu
Department of Computer and Information Science,
University of Pennsylvania
Philadelphia, USA

Hyonyoung Choi
hyonchoi@seas.upenn.edu
Department of Computer and Information Science,
University of Pennsylvania
Philadelphia, USA

Oleg Sokolsky
sokolsky@cis.upenn.edu
Department of Computer and Information Science,
University of Pennsylvania
Philadelphia, USA

Insup Lee
lee@cis.upenn.edu
Department of Computer and Information Science,
University of Pennsylvania
Philadelphia, USA

ABSTRACT

The Internet of Medical Things (IoMT), aided by learning-enabled components, is becoming increasingly important in health monitoring. However, the IoMT-based system must be highly reliable since it directly interacts with the patients. One critical function for facilitating reliable IoMT is anomaly detection, which involves sending alerts when a medical device's usage pattern deviates from normal behavior. Due to the safety-critical nature of IoMT, the anomaly detectors are expected to have consistently high accuracy and low error, ideally being bounded with a guarantee. Besides, since the IoMT-based system is non-stationary, the anomaly detector and the performance guarantee should adapt to the evolving data distributions. To tackle these challenges, we propose a framework for incremental anomaly detection in IoMT with a Probably Approximately Correct (PAC)-based two-sided guarantee, guided by a human-in-the-loop design to accommodate shifts in anomaly distributions. As a result, our framework can improve detection performance and provide a tight guarantee on False Alarm Rate (FAR) and Miss Alarm Rate (MAR). We demonstrate the effectiveness of our design using synthetic data and the real-world IoMT monitoring platform *VitalCore*.

CCS CONCEPTS

• **Computer systems organization** → *Reliability*; • **Human-centered computing** → *Interaction design process and methods.*

KEYWORDS

Internet of Medical Things (IoMT), Anomaly detection, PAC guarantee, Statistical learning.

IoTDI '23, May 09–12, 2023, San Antonio, TX, USA

ACM ISBN 979-8-4007-0037-8/23/05…$15.00
https://doi.org/10.1145/3576842.3582374

ACM Reference Format:
Xiayan Ji, Hyonyoung Choi, Oleg Sokolsky, and Insup Lee. 2023. Incremental Anomaly Detection with Guarantee in the Internet of Medical Things. In *International Conference on Internet-of-Things Design and Implementation (IoTDI '23), May 09–12, 2023, San Antonio, TX, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3576842.3582374

1 INTRODUCTION

Internet of Medical Things (IoMT) is formed with medical devices, embedded software, network capabilities, and physical dynamics of the patient body [15]. Closely monitoring the physiological information of patients, IoMT provides significant benefits for the well-being of patients by increasing the quality of life and cutting medical expenses [2]. With the aging population and increasing number of patients with chronic diseases, we witness an enormous need for IoMT. Since IoMT interacts with the patients directly, the medical community imposes rigorous requirements for its usage. Specifically, IoMT-based systems must be reliable. It should function as expected at all times and not be prone to unexpected failure under normal operating conditions. Besides, the clinicians mandate the reliability of every system component to guarantee the correctness of collected information for diagnostic functions [10].

Therefore, anomaly detection is essential for the reliability of the IoMT-based system. An anomaly detector is tasked with raising alarms when an observation deviates from the normal pattern. For it to be helpful, the classification accuracy should be high, and the error rates should be low [12]. Nevertheless, for a safety-critical system like IoMT, more than average performance is required. There may still be a situation when the error rates suddenly spike, resulting in potentially hazardous patient outcomes. Therefore, a guarantee of the upper bound on the error rates should be in place to assure the system's reliability. Furthermore, for anomaly detection in IoMT, the normal and anomalous patterns evolve dynamically due to the change in usage condition, for example, patient behavioral variation and operational fluctuation [7]. Hence, anomaly detectors in IoMT-based systems should incrementally perform classification with high accuracy and tight guarantee.

We focus on addressing anomaly detection problems in IoMT with evolving usage patterns. For example, as medical technicians encounter more occurrences of regular maintenance, a type of

anomaly that is non-actionable [37], they may no longer perceive the anomaly as anomalous. In other words, although the anomalous pattern persists, they treat it as normal. However, the shift in classification could drastically impair the original anomaly detector's performance and the guarantee's usefulness since they are developed oppositely.

Previous works for incremental anomaly detection span various applications, for example, network intrusion detection [3, 42], forest fire risk prediction [27], airspace operations [9]. However, most of them do not provide performance guarantees in incremental settings. Many uncertainty quantification techniques [1] provide a guarantee. Such techniques often assume a representative calibration data set of the actual data distribution to derive the guarantee [19, 20, 40, 41]. Nevertheless, anomalies could be versatile in practical settings, and the chances are that one calibration set cannot capture all the anomaly distributions. Furthermore, as people observe more incidents of a type of anomaly, the definition of the anomaly may be revised to become normal.

In this work, we propose a framework shown in Figure 1 that provides a Probably Approximately Correct (PAC)-based guarantee for incremental anomaly detection in IoMT. Our framework adopts a human-in-the-loop design, which adapts to the user feedback on the evolving anomaly categories, i.e., expected and unexpected anomalies. With this flexible design, the user can assign the frequently observed anomalies to a calibration set of expected anomalies. Besides, they can progressively expand the unexpected anomaly categories as they discover additional types. As a result, the anomaly detection accuracy is not hampered, along with a confined performance guarantee on False Alarm Rate (FAR) and Miss Alarm Rate (MAR). The two error rates are essential for gauging the anomaly-detecting capability. Miss alarm characterizes missing an actual anomaly, whereas false alarms cause alarm fatigue [30] if there are too many of them. Both have undesirable consequences and, thus, should be minimized for life-critical systems [13] like IoMT.

In summary, our contributions are as follows:

- Propose an incremental framework for detecting expected and unexpected anomalies with guarantee in IoMT.
- Improve the classification accuracy and performance guarantee on FAR and MAR of the underlying anomaly detector.
- Perform an update frequency analysis to show that the framework requires limited user input.
- Evaluate the framework on synthetic data and an IoMT platform (VitalCore) to validate the effectiveness.

The remainder of this paper is structured as follows. First, we start with a literature review in Section 2. Then, we elaborate on the detail of our framework in Section 3 and demonstrate the experimental results of our framework in Section 4. Finally, in Section 5 we discuss the limitation of our framework and conclude the work in Section 6.

2 RELATED WORK

2.1 Incremental Anomaly Detection

Learning-enabled anomaly detectors in IoT need to evolve continuously to adapt to operational variations as new patterns are emerging [17], which is often referred to as incremental anomaly detection [5]. It has broad applications in different domains, for example, network Intrusion Detection Systems (IDSs) [3, 42], system log analysis [4], forest fire risk prediction [27], airspace operations [9, 43], and healthcare [24]. Many online algorithms have been proposed to detect anomalies in ever-changing time series, some have a tree-based structure like Half Space Tree [36], and some are cluster-based with Gaussian Mixture Model (GMM) as the backbone [6, 12, 43]. However, most algorithms do not provide a performance guarantee, which is essential for a life-critical system like IoMT.

2.2 PAC Guarantee

Probably Approximately Correct (PAC) guarantees [20, 40] aim to give a bounded false detection rate for neural networks, based on two user-specified inputs, namely, confidence parameter δ and error parameter ϵ. There are two fundamental false detection error rates in anomaly detection tasks, i.e., FAR and MAR, interchangeably called false-negative and false-positive rates. PAC-Wrap [16] proposes a wrapper around existing anomaly detectors to provide a rigorous PAC guarantee on FAR and MAR. However, there might be multiple anomaly types in practice, for example, expected or unexpected anomalies. Hence, we cannot simply adopt a binary differentiation of anomaly or normality as in [16]. We seek to consider the evolving nature of anomalies and address the problem by adopting a more fine-grained classification of anomalies.

2.3 Dataset Shift Problem

There has been abundant literature studying the dataset shift problem [23], which assumes that the testing data distribution is different from the training data distribution. Some works [28, 33, 34] provide performance guarantees on a more straightforward dataset shift problem — covariate shift problem. It assumes the training input points and test input points follow different distributions. However, the conditional distribution of output values given input points is unchanged. Researchers use the *Importance Weight* [35] to estimate the target distribution from a source distribution and then perform PAC guarantee [21] on top of the estimation.

There is a subtle difference between our problem and dataset shift detection. Firstly, training is not demarcated from testing in our setup since testing instances could be included in the training set for future performance guarantees. Secondly, we assume that there is more than one anomaly distribution. Some of the test time anomalies might follow the same distribution as the training time.

2.4 User-feedback for Recalibration

There are, in general, three ways to perform the recalibration: supervised, semi-supervised, and unsupervised [26, 31, 38, 44]. In our work, we propose to resort to limited user feedback for an update, which may be closest to the semi-supervised definition [11, 18, 25, 32]. A close work [31] also adopt interactive user update to improve detection accuracy. It differs from ours because we aim for high accuracy and, more importantly, a guaranteed error rate. Besides, they leverage two methods to incorporate user updates: metric learning and the Bayesian method. However, the metric learning method [31] is impossible with a vast number of data points. We cannot enumerate all data pairs and instantly compute

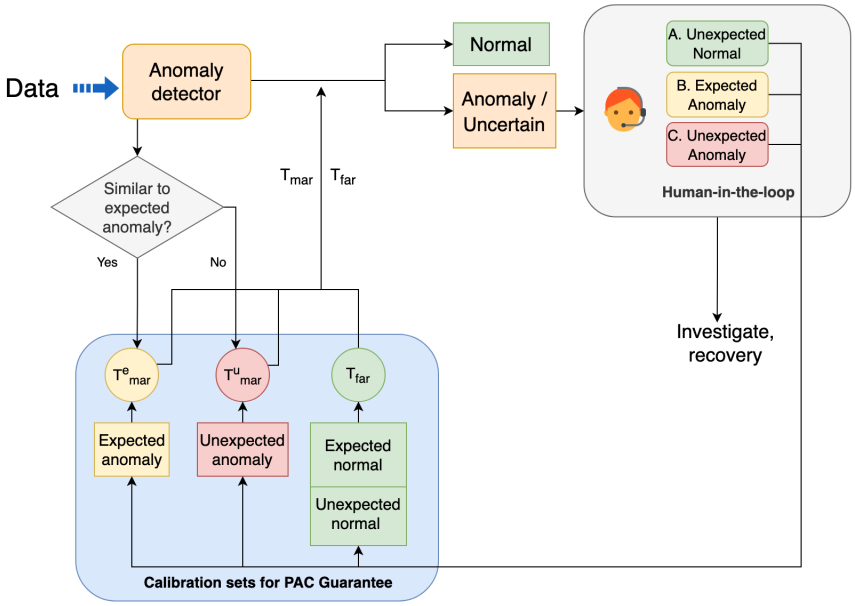


Figure 1: Our framework uses historical data to create calibration sets. New data is fed through an anomaly detector, which compares it to the expected and unexpected anomaly calibration sets. The resulting anomaly score is compared to the thresholds $T_{\mathbf{mar}}$ and $T_{\mathbf{far}}$, which guarantee Miss Alarm Rate (MAR) and False Alarm Rate (FAR), respectively. If the instance is classified as an anomaly or uncertain, human expertise is consulted. The calibration sets are then updated based on user feedback, and new Probably Approximately Correct (PAC) thresholds are computed for the next instance.

the pairwise distance for a large dataset. Hence, we compare our framework with their Bayesian update method in Section 4.3.

3 METHOD

In this section, we describe our framework in detail. First, we formulate the problem of providing a two-sided guarantee. We then explain the PAC guarantee we provide, obtained using the PAC threshold to stratify the anomaly score. After that, we give a motivating example of why we need user feedback to split the anomaly calibration sets into more fine-grained ones. Then, based on the thresholds, we explain how to guarantee FAR and MAR. In addition, we explain how our method can guarantee high accuracy without demanding laborious user input. Finally, we discuss the implementation of our framework.

3.1 Problem Formulation

Let $\mathcal{X}$ be the input space, $\mathcal{Y}$ be the finite label space, and x, y come from the two spaces, respectively; let $\mathcal{D}$ denote a distribution over $\mathcal{X} \times \mathcal{Y}$. We assume a semi-supervised setup with many unlabeled normal instances and a small number of labeled instances. We denote $y = 0$ to be normal and $y = 1$ to be anomalous. Notice that there could be more than one type of anomaly, but to evaluate the anomaly detection accuracy, we treat all of them as $y = 1$. Our goal is to provide a prediction $\hat{y}$ for a test instance x, with FAR and MAR being upper-bounded by an error parameter ϵ. Formally, we want the following:

$$\begin{aligned} \text{FAR} &= \frac{\mathbb{P}_{(x,y)\sim\mathcal{D}}(\hat{y}=1 \mid y=0)}{\mathbb{P}_{(x,y)\sim\mathcal{D}}(\hat{y}=1 \mid y=0)+\mathbb{P}_{(x,y)\sim\mathcal{D}}(\hat{y}=0 \mid y=0)} \\ &= \mathbb{P}_{(x,y)\sim\mathcal{D}_{\text{n}}}(\hat{y}=1) \le \epsilon \end{aligned} \tag{1}$$

where $\mathcal{D}_{\text{n}}$ is the distribution of normal data. Besides, we also want:

$$\begin{aligned} \text{MAR} &= \frac{\mathbb{P}_{(x,y)\sim\mathcal{D}}(\hat{y}=0 \mid y=1)}{\mathbb{P}_{(x,y)\sim\mathcal{D}}(\hat{y}=0 \mid y=1)+\mathbb{P}_{(x,y)\sim\mathcal{D}}(\hat{y}=1 \mid y=1)} \\ &= \mathbb{P}_{(x,y)\sim\mathcal{D}_{\text{a}}}(\hat{y}=0) \le \epsilon \end{aligned} \tag{2}$$

where $\mathcal{D}_{\text{a}}$ is the distribution of anomalous data.

3.2 PAC Thresholds

We adopt the generalization bounds for detection error from [20], which leverages PAC learning theory to construct confidence sets for anomaly detectors with PAC guarantee — i.e., the confidence set for a given input contains the true label with high probability. It is accomplished via constructing the confidence set $C(x)$ based on a one-dimensional parameter T on the probability forecaster $f : \mathcal{X} \mapsto \mathbb{R}$. In detail, we first sort the data in the calibration set according to the score output by f in ascending order, and use the score at position $k^* + 1$ as the threshold, where k^* and the corresponding threshold $\hat{T}$ are calculated as follows:

$$k^* = m\alpha(m, \epsilon, \delta) \tag{3}$$

$$\hat{T} = -\log[f(y_{k^*+1} \mid x_{k^*+1})] \tag{4}$$

Specifically, in the confidence set $C_{\hat{T}}(x)$, we only include the y with a probability greater than $e^{-\hat{T}}$:

$$C_{\hat{T}}(x) = \{y \in \mathcal{Y} \mid f(y \mid x) \geq e^{-\hat{T}}\}.$$

We can treat the anomaly detector as a probability forecaster f. Namely, we provide the PAC guarantee to the anomaly prediction result using the threshold $\hat{T}$ on the anomaly score, computed from an anomaly detector f that forecasts the probability of an instance being anomalous. Formally, given the dataset $(x, y) \sim \mathcal{D}$, a calibration set Z_{cal} with m data points, and $\epsilon, \delta \in \mathbb{R}_{>0}$, we obtain a PAC confidence set $C_{\hat{T}}(x)$ for y, satisfying the guarantee:

$$\mathbb{P}_{Z_{\text{cal}} \sim \mathcal{D}^m}\left[\mathbb{P}_{(x,y)\sim\mathcal{D}}\left(y \in C_{\hat{T}}(x)\right) \geq 1 - \epsilon\right] \geq 1 - \delta. \tag{5}$$

One can always replace the $\mathbb{P}_{(x,y)\sim\mathcal{D}}\left(y \in C_{\hat{T}}(x)\right)$ with other criteria and compute the threshold accordingly. It will then guarantee the corresponding accuracy or error metrics, and we will elaborate on this in Section 3.3.

3.3 Two-sided Guarantee

We train an anomaly detector f on a training set Z_{train} consisting of solely normal instances. We maintain labelled calibration sets $Z_{\text{cal}} = \{Z_n, Z_{a_1}, Z_{a_2}, \ldots, Z_{a_k}\}$, where Z_n means a calibration set for normal data, and $Z_{a_i}, i = 1, \ldots, k$ are the calibration sets for different anomaly types.

To guarantee both the MAR and FAR, the two standard error rates for alarm-issuing applications, we replace the inner part of the formula as in PAC-Wrap [16]. Expressly, on the calibration set consisting of m normal data points Z_{n}, we compute the threshold $\hat{T}_{\text{far}}$ to guarantee FAR:

$$\mathbb{P}_{Z_{\text{n}}\sim\mathcal{D}_{\text{n}}^m}\left[\mathbb{P}_{(x,y)\sim\mathcal{D}_{\text{n}}}(\hat{y} = 1 \mid y = 0) \leq \epsilon\right] \geq 1 - \delta. \tag{6}$$

Similarly, on the anomalous calibration set Z_{a_i} with m data points, we compute the threshold $\hat{T}_{\text{mar}}^{a_i}$ to guarantee MAR on each anomaly type:

$$\mathbb{P}_{Z_{a_i}\sim\mathcal{D}_{a_i}^m}\left[\mathbb{P}_{(x,y)\sim\mathcal{D}_{a_i}}(\hat{y} = 0 \mid y = 1) \leq \epsilon\right] \geq 1 - \delta. \tag{7}$$

According to [20], the thresholds $\hat{T}_{\text{far}}$ and $\hat{T}_{\text{mar}}^{a_i}$ are the solution to Equation (6) and (7). In a high level, it bounds the MAR and FAR below a calibration loss function $\alpha(m, \epsilon, \delta)$, which enforces the ϵ-error and δ-confidence constraint.

Then, we let $\hat{T}_{\text{mar}}$ be the threshold from the closest anomaly calibration set. Together with the threshold $\hat{T}_{\text{far}}$ from the normal calibration set, we can output a guaranteed prediction. Typically, the threshold $\hat{T}_{\text{mar}}$ should lay above $\hat{T}_{\text{far}}$ since the former is calculated from anomalous data that have higher anomaly scores. However, the reverse scenario may occur when the anomalies cannot be easily distinguished from the normal data. We can incrementally relax the ϵ constraint or the δ constraint to allow for a more considerable error margin or lower the confidence until $\hat{T}_{\text{mar}}$ is above $\hat{T}_{\text{far}}$.

Using the two thresholds together, we guide our decision by declaring anything above the $\hat{T}_{\text{mar}}$ to be an anomaly and anything below $\hat{T}_{\text{far}}$ to be normal. Formally, using the two thresholds, we guide our decision for determining anomaly as follows:

$$\hat{y} = \begin{cases} 1 & f(x) \geq \hat{T}_{\text{mar}} \\ \{0, 1\} & \hat{T}_{\text{far}} < f(x) < \hat{T}_{\text{mar}} \\ 0 & f(x) \leq \hat{T}_{\text{far}} \end{cases}. \tag{8}$$

Following this rule, both MAR and FAR will be guaranteed for the anomaly prediction result.

If the anomaly score falls in between the two thresholds, we abstain from making predictions and resort to user feedback in this instance. Ideally, there should not be many instances with an anomaly score between the two thresholds, and the region between the two thresholds is referred to as *uncertainty region*. In Experiment 4.6, we conducted an ablation study to inspect the relationship between the fraction of data points that fall in the uncertainty region and the two user-specified parameters ϵ and δ. Then, if the user demands a concrete decision and $\hat{T}_{\text{mar}}$ is above $\hat{T}_{\text{far}}$, we can use the mean value of the two thresholds as the final threshold to guide our decision, while still maintaining the two-sided guarantee according to [16].

3.4 Fine-grained Anomaly Calibration Sets

As we discussed earlier, the real-world anomaly distribution may be evolving; if we apply a static classification of anomalies, the user would provide the imprecise classification. As a result, the effectiveness of the guarantee we can provide will be hamstrung. An illustrating example is as follows.

For the IoMT we monitor using VitalCore, the maintenance would suspend the system and trigger an anomalous pattern of disconnection, which is observed as a spike in time interval between two consecutive messages. The pattern is very different from normal patterns, which have a consistent time interval of around 60 seconds between two messages. Hence, the maintenance is predicted as an anomaly by the anomaly detector. Since we do not have the up-to-date maintenance schedule, we can not remove the maintenance data. Besides, the technicians want to keep the maintenance data to confirm that the maintenance happens as expected. We prompt the user to decide on the category for the maintenance data. Initially, we apply a static classification of anomalies, maintaining a single anomaly calibration set and a normal calibration set. After seeing some maintenance instances, the user regards them as expected and prefers not to be bothered by the alerts on such events. As a result, the user assigns maintenance instances to the normal calibration instead of the anomalous one. However, this assignment contaminates the normal training and calibration set by mixing different data distributions, disabling us from providing a high classification accuracy and a meaningful guarantee, as we show in Experiment 4.2 and Experiment 4.3.

We can avoid the trivial guarantee by modifying the original classification criteria to adapt to the change. In other words, instead of predicting an instance to be either anomaly or a normal instance, we incorporate the user's perception and create a new class of anomaly — expected anomaly. Although the users are not directly involved in calibration process, they are prompted to provide labeling on anomalies and uncertain examples. For anomalies caused by maintenance, we include them in the newly created calibration set for the expected anomaly. The adjustment in classification criteria

might affect the existing calibration set. The historical calibration set could be updated by migrating or deleting the records to reflect the change. Compared with a single anomaly type, the fine-grained calibration sets with more anomaly types significantly improve the precision of the guarantee. We illustrate this example with Experiment 4.3.

We compute the PAC threshold for each fine-grained anomaly calibration set a_i. Then, when an instance arrives, we choose the most appropriate anomaly calibration set $\hat{a}_i$ for it by taking the one with minimal Euclidean distance between the instance and the centroid of the anomaly calibration set.

$$\hat{a}_i = \underset{a_i}{\operatorname{argmin}} \sqrt{(x - \mu_{a_i})^2} \tag{9}$$

If there is more than one calibration set with the same minimal Euclidean distance, we choose the one with a smaller index. A side benefit of splitting the calibration set is the reduction in computation time of the PAC thresholds, which scales almost linearly with the calibration set size, as we will show in Experiment 4.5. In addition, since the threshold of different anomaly types are independent of each other, we can conduct the calculation parallelly on different machines, increasing the computational efficiency. Although we illustrate our result with two anomaly types, namely, the expected anomaly and the unexpected anomaly, there could be more fine-grained anomaly sub-types in practice. For example, we can treat different attack types as the sub-types of unexpected anomalies in the network intrusion detection [14]. There could be multiple sub-types of anomalies as the user defines, for example, Denial of Service (DoS) attack, R2L attack, U2R attack, probing attack and many more. Our framework can be flexibly generalized to multiple anomaly sub-types, as illustrated in Experiment 4.5.

3.5 Update frequency

Suppose the prediction on an instance turns out to be anomalous or uncertain. In that case, we seek help from the user, with the options of approving or modifying the current label of the anomaly instance. Note that we only update the calibration set with the labeled instances. It is natural to question the practicability if a system frequently requires the user to provide feedback. Especially in the medical domain, clinicians and technicians are concurrently tasked with numerous monitoring duties. Fortunately, our framework only requires infrequent user feedback to generate a stable PAC guarantee.

To start with, suppose the user is busy and can only respond to a limited number of alerts. Specifically, for every c alert(s), the user provides a label for any of them and misses the rest. The update frequency (F) is then defined as the inverse of the number of alerts generated and: $F = \frac{1}{c}$. Let t be the total number of alerts generated in the period we monitor. We have a labeled calibration set of the size: $m = m_0 + Ft = m_0 + \frac{t}{c}$, where m_0 is the initial calibration set size. Notice that $F = 0$ is defined as no update, and we use the maximal anomaly score from the training set as the threshold.

Update frequency affects the guarantee via the calibration loss. Specifically, we have a larger calibration set as we update more frequently. The increased calibration set size leads to a larger allowed calibration loss, denoted as $\alpha(m, \epsilon, \delta)$. This is because $\alpha(m, \epsilon, \delta)$ is

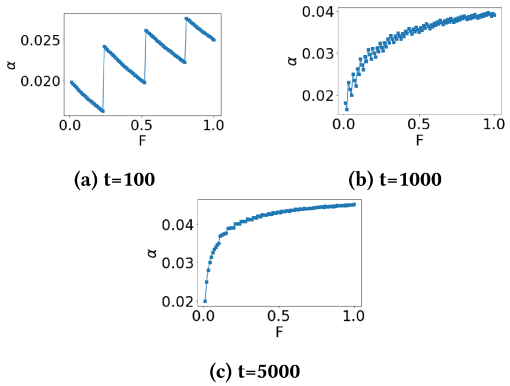


Figure 2: Relationship between F and α.

an increasing function of calibration set size m [20]:

$$\alpha(m, \epsilon, \delta) = \epsilon - \sqrt{\frac{\log(2m) + 1 - \log(\delta/4)}{m}}. \tag{10}$$

At first sight, the high frequency increasing the calibration loss might seem counter-intuitive since we normally expect increasing the effort to result in something beneficial. However, it should be alternatively interpreted as raising the selectiveness of $C_{\hat{T}}(x)$. As we have a higher α, we have a larger k^* by Equation (3). Since we sort the anomaly score $f(y_k \mid x_k)$ in ascending order, we have a smaller $\hat{T}$ and larger $e^{-\hat{T}}$ with a larger k^* by Equation (4). With a larger threshold $e^{-\hat{T}}$, we include less label y to the confidence set $C_{\hat{T}}(x)$ on average. Therefore, a high update frequency is favorable because it creates a more refined confidence set, which reduces the likelihood of getting a trivial $C_{\hat{T}}(x)$ containing all the labels in $\mathcal{Y}$, i.e., $C_{\hat{T}}(x) = \{0, 1\}$.

Moreover, there is a decreasing marginal effect in the update frequency or, equivalently, the calibration set size. Taking a first-order derivative of Equation (10), we get:

$$\frac{d\alpha(m, \epsilon, \delta)}{dm} = \frac{\log(2m)}{2m^2\sqrt{\frac{\log(2m)+1-\log(\delta/4)}{m}}} \tag{11}$$

The derivative in Equation (11) is positive for $m \geq 1$ and then converges to zero as we increase m. Specifically, as we update more frequently, the confidence set shrinks less beyond a point. We visualize the relationship between frequency F and the α in Figure 2, given different numbers of total alerts: $t = [100, 1000, 5000]$ with a initial calibration set size $m_0 = 100$, and $\epsilon = \delta = 0.05$. We found that the PAC confidence set is getting more selective (α getting closer to $\epsilon = 0.05$) as we update more frequently. Nevertheless, in the short run, as shown in Figure 2a and 2b, we could end up with a locally lower α as we update more frequently. According to Equation (8) in [20], larger m leads to a bigger binomial sum initially, and to satisfy the δ constraint, we must choose smaller k and hence lower α. Overall α increases and the PAC confidence set gets refined as we increase the update frequency F. As shown in Figure 2c, our framework has a decreasing return on margin regarding update frequency with $t = 5000$. Therefore, our method does not require the user to respond to every alert generated; instead, it is sufficient to label with a frequency generating the calibration loss close to

ϵ. To evaluate our framework more straightforwardly, we report in Experiment 4.5 the relationship between update frequency and final accuracy of the confidence set, which ideally should have a similar pattern to that of the α.

3.6 Implementation

Two computations are involved in providing the guarantee: the threshold computation on calibration sets and the inference on the test instance. The testing inference is a binary operation, i.e., comparing the anomaly score of the instance against the thresholds. It takes around the granularity of $10^{-5}s$ on most devices, which is undoubtedly feasible for practical deployment. The focus is on capping the size of the calibration set to compute the PAC threshold efficiently.

The computation complexity of PAC is $O(m)$, i.e., it scales linearly with the calibration set size, as we will show in Experiment 4.5. We obtain smaller calibration sets as a side benefit using fine-grained calibration sets. Besides, as we explicate the decreasing marginal effect of calibration set size in Section 3.5, the PAC guarantee does not necessitate unreasonably large data size. Users may shorten the computation time for calculating the threshold to suit their needs, as long as a minimum size of:

$$m = \frac{\log \delta}{\log(1-\epsilon)}$$

is kept for the calibration set size. This is as little as $m = 59$ for $\epsilon = \delta = 0.05$.

We implement the our framework with a record pool to reflect the update frequency F. For every $\frac{1}{F}$ records accumulated in the pool, an alert is issued to the user and requests for feedback. The user can choose to provide feedback to one or multiple of the records. The ones with feedback are added to the corresponding calibration sets. The description of our framework is in Algorithm 1.

Algorithm 1 PAC guarantee for evolving data

Input: anomaly detector f, instance x, error level ϵ, confidence level δ, the user update frequency F.
Output: anomaly prediction $\hat{y}$
Compute $\hat{T}_{\text{far}}$ according to Equation (3), (4) and (6).
Compute the $\hat{T}_{\text{mar}}$ from the closest anomaly calibration set according to Equation (3), (4), (7), and (9).
$\hat{y}$ is determined according to Equation (8).
count = 0, pool = [ ].
if $\hat{y} \neq 0$ **then**
  pool.append(x).
  count = count + 1.
  if count = $\frac{1}{F}$ **then**
    Issue an alert to the user for feedback.
    if user provide feedback $y' = i$ (or unexpected normal $y' = 0$) **then**
      Add x' to Z_{a_i} (or Z_n).
    end if
    count = 0, pool = [ ].
  end if
end if
return $\hat{y}$

4 EXPERIMENTAL RESULTS

We identified the below questions to validate the effectiveness of our framework:

- **Q1 Detector Improvement**: how can the underlying anomaly detectors benefit from the incremental anomaly types?
- **Q2 Adaptive recalibration**: what is the performance on the synthetic and real-world dataset using the adaptive PAC calibration sets?
- **Q3 Update frequency**: How many alerts does the user need to provide a label for real scenarios?
- **Q4 Time Complexity**: How is the computation time (in wall clock seconds) scale with the number of anomaly types and calibration set size?
- **Q5 Ablation Study**: How is ϵ, δ going to affect the size of the uncertainty region?

4.1 Experimental Setup

4.1.1 Dataset. **Synthetic data set:** We generate the synthetic dataset with a total of 15000 data points from three 6-dimensional normal distributions N_1, N_2, N_3 with the same covariance matrix but with different means $\mu_1, \mu_2, \mu_3 \in \mathbb{R}^6$. Let I_p be the p-dimensional identity matrix with $p = 6$, and σ^2 be a uniformly random value drawn over $[1, 100]$. *Python* sklearn.datasets.make_classification library is used. We treat N_1 as the normal distribution, N_2 as expected anomalous distribution and N_3 as unexpected anomalous distribution. We have:

$$X_{\text{normal}} \sim \mathcal{N}(\mu_1, \sigma^2 I_p)$$
$$X_{\text{expected anomalous}} \sim \mathcal{N}(\mu_2, \sigma^2 I_p)$$
$$X_{\text{unexpected anomalous}} \sim \mathcal{N}(\mu_3, \sigma^2 I_p).$$

VitalCore dataset: We experiment on a 6-dimensional real-world data set that monitors the IoMT usage patterns collected on the VitalCore platform. It consists of over 3000 medical devices, and we record their connection status at the granularity of one minute. We extract six features from the records: month, day, hour, day of the week, whether in a business hour, and the interval between two consecutive records. The data we collected has three usage patterns: the connected pattern with a one-minute interval (normal), the regular maintenance pattern (expected anomaly), and the network outage pattern (unexpected anomaly). These patterns are obtained with the labels provided by the technicians. We look at the time series with a sliding window of 30 minutes upon getting the data, and the count for the number of sliding window sequences in each category is:

- Normal: 418523
- Expected anomaly: 512
- unexpected anomaly: 4257

We may vary the anomaly ratio in the data to study the effectiveness of our guarantee.

4.1.2 Anomaly Detector. We employ an anomaly detector to calculate a 1-dimensional anomaly score for computing the PAC thresholds. On the synthetic data, we used a simple anomaly detector One-class Support Vector Machine [29]. On the VitalCore data, we adapt from an AutoEncoder-based anomaly detector [39], which

has the best empirical prediction accuracy on VitalCore data [8]. Notice that the choice of anomaly detector is not the focus of our work since our framework is model-agnostic. It provides a two-sided guarantee for virtually any existing anomaly detector that can compute the anomaly score.

4.1.3 Metrics. We check whether the estimated FAR and MAR defined in Equation (1) and (2) are below the specified ϵ constraint. Since anomaly detection is a binary classification problem, we consider all anomaly classes as one and the normality as zero. We conduct 10 Monte Carlo trials for all experiments and report the averaged result with statistical significance computed at 95% confidence level.

In Experiment 4.2, we report the Area Under the Receiver Operating Characteristic Curve (ROCAUC) and the Precision-Recall Area Under Curve (PRAUC) Score. The ROC curve is the plot of the False Positive Rate (FPR) (in the x-axis) versus the True Positive Rate (in the y-axis) across all thresholds. ROCAUC computes the Area under the ROC curve, a standard metric for comparing binary classifier models directly. However, ROC curves may provide an excessively optimistic view of the performance for imbalanced binary classification; researchers also refer to the PRAUC for a more comprehensive comparison. A Precision-Recall curve (or PR Curve) is a plot of the recall (in the x-axis) and the precision (in the y-axis) for different probability thresholds. The PR curve focuses on the minority class, making it an effective diagnostic for imbalanced binary classification models like anomaly detectors. Similarly, PRAUC summarizes the PR curve with a range of threshold values as a single score.

4.1.4 Configuration Details. Our framework is implemented using PyTorch [22]. All experiments, including timings, were run with 4 Nvidia 2080Ti GPU, 80 vCPUs, a processor Intel(R) Xeon(R) Gold 6148 @ 2.4 GHz and 768GiB of RAM.

4.1.5 Baseline. An alternative way to our way of updating the calibration set is using a Bayesian approach to update the posterior as in [31]. We could adjust the probability of an instance belonging to each anomaly type as we receive user feedback. Let $P(A \mid B)$ be the posterior probability we want to update, where A denotes a data point as an anomaly, and B denotes that it is predicted as normal. Then, we follow the setup in [31] and update the probability $p = P(A)$ whenever we receive an update from the user. Besides, we uniformly decide the environment complexity parameter $q = P(B \mid A) = 1/3$. The posterior probability p is updated as follows:

$$p = \frac{P(B \mid A)P(A)}{P(B)} = \frac{qp}{1 - p(1-q)}$$

In other words, as we receive more normal feedback from the user, we decrease the posterior probability of an instance being an anomaly. We conduct the update for the posterior of the normal whenever it is an anomaly update, whether expected or unexpected. Besides, we normalize the probability with a softmax function after each update to ensure it always remains in the $[0, 1]$ range.

4.2 Q1 Detector Improvement

The classification accuracy of the anomaly detectors improves as they receive incremental anomaly categorization. We illustrate this by comparing the ROCAUC and PRAUC performance of the anomaly detectors with and without fine-grained anomaly distinction. We first train an anomaly detector on a training set of 5000 data points. The expected anomaly is considered normal if we have a single anomaly type. Only unexpected anomalies are considered anomalies. On the other hand, if we have two anomaly types, we distinguish expected anomaly from normality and train with only normal data. Specifically, on synthetic data, we have:

- **Single anomaly type:** the training set contains 2500 data points from N_1 and 2500 data points from N_2.
- **Two anomaly types:** the training set contains 5000 data points from N_1.

On VitalCore data, we have:

- **Single anomaly type:** the training set contains 2500 data points with the normal connected pattern and 2500 data points with the regular maintenance pattern.
- **Two anomaly types:** the training set contains 5000 data points from the normal connected pattern.

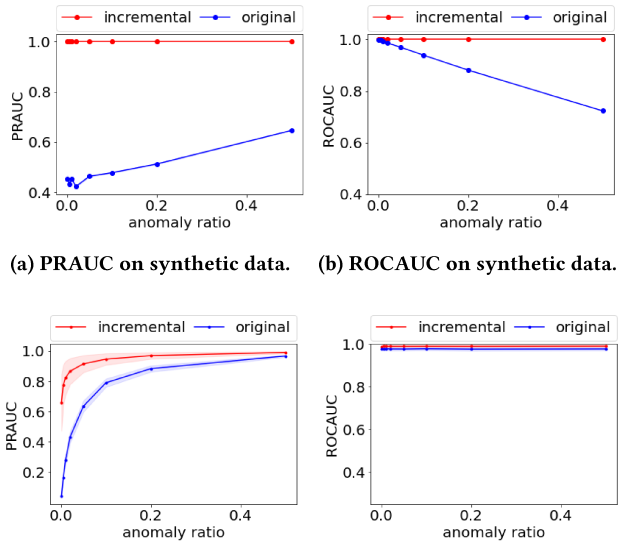
On the testing set with 5000 data points, we randomly shuffle all data points from the three distributions and evaluate the ROCAUC and PRAUC.

Using a single anomaly type categorization hurts the performance of the anomaly detector. An anomaly detector usually learns a homogeneous normal pattern. It classifies any data points that deviate from the pattern as an anomaly. The "contaminated" normal category confounds the anomaly detector with heterogeneous patterns. The detection performance is shown in Figure 3a, 3b for synthetic data and Figure 3c, 3d for VitalCore data. To validate the effect of contamination, we experiment with different anomaly ratios a. We present the original anomaly detector performance with the blue lines labeled "original" and the one with incremental anomaly categorization using the red lines labeled "incremental".

As shown in Figure 3, incremental anomaly detectors have higher ROCAUC and PRAUC. It indicates that the detection performance improves with the fine-grained classification of expected anomalies. On the synthetic data, we find the PRAUC of the original anomaly detector is close to 0.5. Therefore, mixing the expected anomaly with normal data would significantly impact the classification performance. The gap in the original ROCAUC and PRAUC reveals the imbalanced data problem. Even though we have a high ROCAUC (close to 1.0) with a small anomaly ratio, the PRAUC is low (less than 0.5). Hence, we should consider both metrics to evaluate the anomaly detection performance. As a result, adopting the fine-grained distinction would improve the PRAUC by 0.52 and the ROCAUC by 0.06 on average across different anomaly ratios.

On the VitalCore data, the PRAUC increases by 0.35, and the ROCAUC increases by 0.01 on average. In addition, we can see that the PRAUC falls below 0.5 when the anomaly ratio is smaller than 5%, indicating that the anomaly detector performs no better than random guessing on anomalies. However, a minor anomaly ratio is usually the case in reality. Fortunately, our framework can significantly improve the anomaly detector performance to have an average PRAUC greater than 0.65, even with a minuscule anomaly ratio like 0.1%.

Moreover, the PRAUC improvement is not as significant as the synthetic data when we have a large anomaly ratio above 5%. It



(a) PRAUC on synthetic data. (b) ROCAUC on synthetic data.

(c) PRAUC on VitalCore data. (d) ROCAUC on VitalCore data.

Figure 3: The PRAUC and ROCAUC performance improves with the incremental distinction of anomaly types.

may be because the expected anomaly pattern in VitalCore is close to the normal data. Expected anomalies are system reboots that usually recover within three minutes, and normal patterns are consistent at one-minute intervals. However, a network outage could last up to several hours. Hence, the anomaly detector may merge the expected anomaly and normal into a single cluster and classify them against the unexpected anomaly well. Despite this, the anomaly detector would still benefit from an updated anomaly categorization, especially when we have a relatively small anomaly ratio.

4.3 Q2 Adaptive recalibration

To evaluate the effectiveness of our guarantee, we compare the performance with and without adaptive recalibration. Specifically, without recalibration, we have a single anomaly-type calibration set. However, with the adaptive change to accommodate evolving anomalies, we have two fine-grained anomaly calibration sets, i.e., expected anomaly and unexpected anomaly. Besides, we also compared our update method with a Bayesian update approach in [31].

4.3.1 Synthetic data. On the synthetic data, we first train an anomaly detector on a training set of 5000 data points drawn from N_1. Then, we simulate real-world settings to feed data into the system. We use the trained anomaly detector to compute the anomaly score on the calibration set with 5000 data points. On the anomaly score computed from the calibration set, we calculate the two thresholds:

- **Single anomaly type:** we calculate threshold $\hat{T}_{\text{far}}$ from the "normal" calibration set consists of half of N_1 and half of N_2, and calculate $\hat{T}_{\text{mar}}$ on the anomaly score of N_3.
- **Two anomaly types:** we calculate threshold $\hat{T}_{\text{far}}$ on N_1, and calculate $\hat{T}^{a_1}_{\text{mar}}$ and $\hat{T}^{a_2}_{\text{mar}}$ on N_2 and N_3 respectively.

On the testing set with 5000 data points, we randomly shuffle all data points from the three distributions. For each point fed into the system, we get the anomaly score from the trained anomaly detector and use the thresholds to guide our detection:

- **Single anomaly type:** we use $\hat{T}_{\text{far}}$ and $\hat{T}_{\text{mar}}$ as calculated above.
- **Two anomaly types:** we calculate the Euclidean distance of a data point to the calibration set centroid of N_2 and N_3 as in Equation (9). We then use the threshold of the closer one as the $\hat{T}_{\text{mar}}$ together with $\hat{T}_{\text{far}}$ to determine an anomaly.

Eventually, we evaluate the final FAR and MAR on the testing set. We experiment with different levels of anomaly ratio $a = [0.1\%, 0.5\%, 1\%, 2\%, 5\%, 15\%, 20\%, 50\%]$ while fixing the calibration set size to be 5000 and see how the PAC guarantee is affected. Initially, we set the error constraint to be $\epsilon = 0.02$. Then, if it cannot be satisfied, we increment at a step of 0.1 each round until both error rates can be guaranteed below the updated ϵ.

As a result, if we do not adaptively include new anomaly types, the guarantee we can provide is imprecise. The results are shown in Table 1, a denotes the anomaly ratio, ϵ is the guaranteed upper bound for error rates, U_0 is the initial uncertainty region without relaxing ϵ, and U is the final uncertainty region. With a single anomaly type, the error rate we can guarantee goes from 0.02 to 0.62 with an increasing ratio of anomaly. Since column MAR and FAR are smaller than column ϵ with 95% confidence, the guarantee is satisfied. However, at the level of 0.62, we can only guarantee that there would be around half the chance that the alarm is not a false alarm or that we will not miss an actual alarm, which is of limited usefulness. Figure 4a shows the anomaly score distribution. We can see that the normal calibration set contains both real normal instances and anomalous instances that are perceived as normal; thus, the maximal anomaly score for the normal calibration set is high.

Also, without relaxing the ϵ constraint, the uncertain region between the two thresholds contains more and more data points, i.e., from 23% to 89%, as shown in column U_0 of Table 1. Intuitively, it means that with more anomalies mixed up in the normal calibration set, we get more confused and abstain from predicting at the initial level of $\epsilon = 0.02$, which aligns with our expectations. Therefore, we relax the error constraint incrementally to reduce the uncertainty region. As a result, the final uncertainty region in U is much smaller than U_0.

On the other hand, with the fine-grained calibration sets, the ϵ guarantee we can get is precise, i.e., $\epsilon = 0.02$. The result is shown in columns ϵ, MAR_{pac} and FAR_{pac} of Table 2, implying that using the user perception to create fine-grained anomaly calibration sets can significantly improve the guarantee we can provide to the user. Furthermore, the uncertain region U_0, U between the two thresholds is consistently lower than 2%. It illustrates that we are relatively sure about the prediction with the fine-grained anomaly calibration sets. Hence, we do not abstain from making predictions on more than 2% of the test data. The anomaly score distribution for the fine-grained calibration sets is shown in Figure 4b, and the normal calibration set contains only the actual normal instances.

a	ϵ	MAR	FAR	U_0	U
0.1%	0.02	0.0 ± 0.0	0.0015 ± 0.0	0.0191 ± 0.0007	0.0191 ± 0.0007
0.5%	0.02	0.0 ± 0.0	0.0062 ± 0.0001	0.0134 ± 0.0006	0.0134 ± 0.0006
1%	0.02	0.0 ± 0.0	0.0123 ± 0.0001	0.0057 ± 0.0007	0.0057 ± 0.0007
5%	0.22	0.1297 ± 0.0025	0.2066 ± 0.0003	0.2291 ± 0.0004	0.0128 ± 0.0004
10%	0.42	0.2897 ± 0.0037	0.3409 ± 0.0007	0.4004 ± 0.0005	0.073 ± 0.0013
15%	0.52	0.3781 ± 0.0021	0.4239 ± 0.0009	0.5265 ± 0.0005	0.1464 ± 0.0011
20%	0.52	0.4371 ± 0.0027	0.4826 ± 0.0012	0.6234 ± 0.0006	0.0743 ± 0.0025
50%	0.62	0.6039 ± 0.0011	0.5921 ± 0.0009	0.8935 ± 0.0007	0.0359 ± 0.0025

Table 1: Single-anomaly-type calibration set for synthetic data.

a	ϵ	MAR_{pac}	$\text{MAR}_{\text{bayes}}$	FAR_{pac}	$\text{FAR}_{\text{bayes}}$	U_0	U
0.1%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0004 ± 0.0	0.0176 ± 0.0004	0.0176 ± 0.0004
0.5%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0004 ± 0.0	0.0182 ± 0.0005	0.0182 ± 0.0005
1%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0004 ± 0.0	0.0171 ± 0.0006	0.0171 ± 0.0006
2%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0005 ± 0.0	0.0183 ± 0.0003	0.0183 ± 0.0003
5%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0007 ± 0.0	0.0171 ± 0.0004	0.0171 ± 0.0004
10%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0006 ± 0.0	0.0176 ± 0.0004	0.0176 ± 0.0004
15%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0001 ± 0.0	0.0163 ± 0.0004	0.0163 ± 0.0004
20%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0002 ± 0.0	0.0167 ± 0.0002	0.0167 ± 0.0002
50%	0.02	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0001 ± 0.0	0.0136 ± 0.0002	0.0136 ± 0.0002

Table 2: Two-anomaly-type calibration sets for synthetic data, compared with the Bayesian approach. MAR_{pac} and FAR_{pac} denotes the MAR and FAR using the our update approach. $\text{MAR}_{\text{bayes}}$ and $\text{FAR}_{\text{bayes}}$ denotes the MAR and FAR using the Bayesian approach.

The Bayesian approach, which is shown in the $\text{MAR}_{\text{bayes}}$ and FAR_{pac} columns of Table 2, also meets the initial $\epsilon = 0.02$ guarantee. However, our framework has a lower FAR and a comparable MAR.

4.3.2 VitalCore data. On VitalCore data, we follow the same training, calibrating, and testing procedure and data set size as the one on synthetic data. Furthermore, we observe a similar result for contaminating the normal calibration set with the expected anomalies. The results are shown in Table 3, 4, and Figure 5. Using two-anomaly-type calibration sets separates the expected anomaly from normal. It enables us to have a clean normal score distribution, as shown in Figure 5b. As a result, we have a more precise guarantee based on column ϵ of Table 3 and Table 4, i.e., from 0.12 to 0.09. However, the improvement is not as significant as in the synthetic data for a similar reason as we discussed in Experiment 4.2. Despite this, it is still advantageous to use the fine-grained calibration sets for tight control of both MAR and FAR. The uncertainty region U_0 also expands as we increase the anomaly ratio, mitigated as we relax the error constraint, as shown in column U.

The result for using the Bayesian approach on VitalCore data is shown in column $\text{MAR}_{\text{bayes}}$ and $\text{FAR}_{\text{bayes}}$ of Table 4. It manifests the benefit of having a guarantee for error rates. Although the Bayesian method generates a smaller MAR than ours, it has FAR of around 40%, violating the $\epsilon = 0.09$ constraint on FAR. The high FAR would lead to the alarm fatigue phenomenon among clinicians. On the other hand, our framework ensures that both FAR and MAR are below the $\epsilon = 0.09$ constraint, which is shown in columns MAR_{pac} and FAR_{pac} of Table 4.

4.4 Q3 Update frequency

To free the user from continual labeling of the alert generated, we demonstrate that our framework does not require a high update frequency (e.g., labeling every alert). In addition, it would be sufficient for the user to label at a frequency that gives a high accuracy close to the convergence accuracy.

To validate our theoretical analysis on an actual application, we experiment with the VitalCore data from April to October 2022. During these six months, the platform generates 146 alerts for 22 servers across ten hospitals. The technicians provide labels for 130 unexpected anomalies. We begin with an initial calibration set of the size $m_0 = 2500$, and $\epsilon = \delta = 0.05$. Then, we add labeled instances to the calibration set. During expanding calibration sets, we vary the update frequency to study its effect on accuracy. Then, we evaluate the resultant accuracy on the testing set; we plot around it with a 95% confidence interval.

The result in Figure 6 confirms our theoretical analysis. As the update frequency reaches approximately $F^* = 0.27$, equivalent to 35 alerts in real life, the accuracy stays close to perfect accuracy. Hence, there is no incentive to increase the update frequency further and provide labels to all 146 alerts. The update frequency suggests 35 anomaly labels over six months, which is approximately one label input for the anomaly per week. Moreover, the time needed to converge to a high accuracy depends on the user-specified error and confidence level. The higher the level (i.e., more relaxed), the faster we can gather enough feedback from the user and guarantee accurate performance.

a	ϵ	MAR	FAR	U_0	U
0.1%	0.12	0.0464 ± 0.0312	0.0629 ± 0.0213	0.1121 ± 0.0636	0.0658 ± 0.022
0.5%	0.12	0.053 ± 0.0301	0.0604 ± 0.0184	0.1279 ± 0.0799	0.0699 ± 0.0146
1%	0.12	0.0533 ± 0.0313	0.0566 ± 0.0164	0.1177 ± 0.0566	0.0776 ± 0.0126
2%	0.12	0.0468 ± 0.0236	0.0577 ± 0.0176	0.1126 ± 0.071	0.0735 ± 0.0144
5%	0.12	0.0456 ± 0.0237	0.0598 ± 0.0191	0.1458 ± 0.0792	0.0663 ± 0.0184
10%	0.12	0.0498 ± 0.0246	0.0596 ± 0.0189	0.1302 ± 0.0678	0.0737 ± 0.0186
15%	0.12	0.051 ± 0.0263	0.06 ± 0.0187	0.127 ± 0.0552	0.0756 ± 0.0173
20%	0.12	0.0397 ± 0.0209	0.0641 ± 0.0209	0.1492 ± 0.0617	0.0691 ± 0.0197
50%	0.12	0.0579 ± 0.0307	0.0552 ± 0.0159	0.2221 ± 0.0563	0.0908 ± 0.014

Table 3: Single-anomaly-type calibration set with VitalCore data.

a	ϵ	MAR_{pac}	$\text{MAR}_{\text{bayes}}$	FAR_{pac}	$\text{FAR}_{\text{bayes}}$	U_0	U
0.1%	0.09	0.0504 ± 0.0278	0.0 ± 0.0	0.0419 ± 0.0225	0.4051 ± 0.0	0.0942 ± 0.0574	0.064 ± 0.0274
0.5%	0.09	0.0398 ± 0.0213	0.0 ± 0.0	0.0465 ± 0.0259	0.4040 ± 0.0	0.1348 ± 0.0892	0.0591 ± 0.0269
1%	0.09	0.0432 ± 0.0219	0.0 ± 0.0	0.0419 ± 0.0233	0.4060 ± 0.0	0.1057 ± 0.0708	0.0636 ± 0.0267
2%	0.09	0.0416 ± 0.0219	0.0 ± 0.0	0.044 ± 0.0244	0.4060 ± 0.0	0.0973 ± 0.0726	0.0614 ± 0.0268
5%	0.09	0.0387 ± 0.0204	0.0 ± 0.0	0.0485 ± 0.0283	0.4048 ± 0.0	0.1395 ± 0.0948	0.0554 ± 0.0291
10%	0.08	0.0361 ± 0.0193	0.0 ± 0.0	0.0439 ± 0.0245	0.4060 ± 0.0	0.1074 ± 0.076	0.0491 ± 0.0256
15%	0.09	0.0422 ± 0.0222	0.0 ± 0.0	0.0422 ± 0.0239	0.4039 ± 0.0	0.1283 ± 0.0861	0.0632 ± 0.0266
20%	0.09	0.0453 ± 0.0238	0.0 ± 0.0	0.0405 ± 0.0223	0.4070 ± 0.0	0.0928 ± 0.0626	0.0636 ± 0.027
50%	0.09	0.0396 ± 0.0221	0.0 ± 0.0	0.0420 ± 0.0228	0.4056 ± 0.0	0.0918 ± 0.0614	0.0597 ± 0.024

Table 4: Two-anomaly-type calibration set with VitalCore data, compared with the Bayesian approach. MAR_{pac} and FAR_{pac} denotes the MAR and FAR using the our update approach. $\text{MAR}_{\text{bayes}}$ and $\text{FAR}_{\text{bayes}}$ denotes the MAR and FAR using the Bayesian approach.

4.5 Q4 Computation Time

In this experiment, we study the scalability of our framework with different calibration set sizes and anomaly types. The experiment of varying anomaly types considers the potential distribution shift in the future, where the unexpected anomalies further evolve into more anomaly sub-categories. We recorded the time needed to calculate the thresholds on the calibration set (*calibration*) as well as the inference time during testing (*test*). Specifically, we change the calibration set size and number of anomaly sub-types to see how our framework scales with them.

We first describe how we generate the data for Table 5 and Table 6. To generate Table 5, we vary the calibration set size from 2000 to 20000, with anomaly ratio $a = 0.05$. For Table 6, we generate from one to ten unexpected anomaly sub-types. Specifically, we sample data from different normal distributions with distinct means $\mu_2, \mu_3, \ldots, \mu_{11}$. Meanwhile, we fix the calibration size to be 10000 to have sufficient instances for each anomaly type. In Table 6, we start with two anomaly types in the first row (expected and unexpected) and then add one more unexpected sub-type for the next row, and so on.

In Table 5 and Table 6, we can see that the time needed to compute the threshold scales almost linearly with calibration size and the number of anomaly types. The inference time for each instance takes around 10^{-5} seconds, making it feasible for real-time applications. As we increase the calibration set size, we expect the *test* to be relatively constant. However, the *test* time also increases, as shown in column *test* of Table 5. It may be because we need more time to compute the centroid of a larger calibration set.

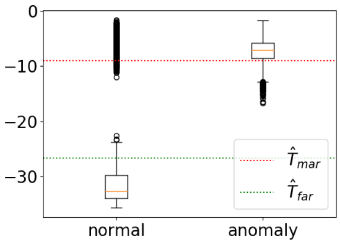
size	*calibration*	*test*
2000	0.8858 ± 0.0104	$7.2519 \times 10^{-6} \pm 9.0409 \times 10^{-8}$
3000	1.3554 ± 0.0192	$7.2484 \times 10^{-6} \pm 8.9195 \times 10^{-8}$
5000	2.6700 ± 0.0297	$9.6435 \times 10^{-6} \pm 6.8809 \times 10^{-7}$
10000	5.0126 ± 0.0523	$1.0513 \times 10^{-5} \pm 8.6625 \times 10^{-7}$
15000	7.2046 ± 0.0247	$1.0234 \times 10^{-5} \pm 9.4843 \times 10^{-7}$
20000	10.1308 ± 0.2232	$1.0529 \times 10^{-5} \pm 8.5205 \times 10^{-7}$

Table 5: *Calibration* and *test* time with different calibration set *size*.

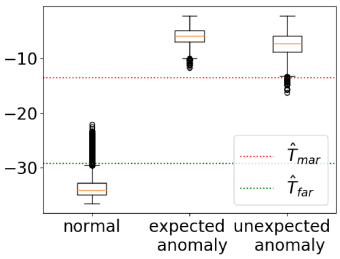
Furthermore, since the PAC threshold computations of different anomaly types are independent, we can distribute the workloads to several machines to curtail the linear growth. The result is shown in column *parallel* of Table 6. However, the benefit of parallelization is not very obvious when we have less then nine anomaly types due to the computation overhead.

4.6 Q5 Ablation Study

Figure 7 shows the mean uncertainty region with different values of user-specified ϵ and δ. It is computed on VitalCore data with its original anomaly ratio $a = 0.05$, and we perform no recalibration. The uncertainty region increases as we impose a more stringent requirement on error and confidence levels, revealing a trade-off

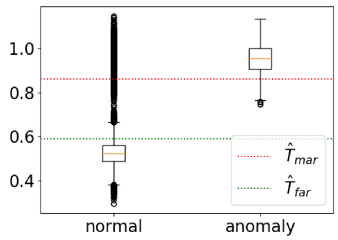


(a) With a single anomaly calibration set, the normal calibration set is contaminated by anomalies that are perceived as normal. Hence it has two discrete anomaly score distributions.

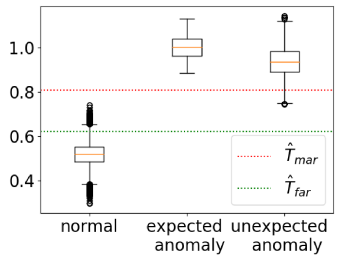


(b) With the fine-grained calibration sets, we can distinguish normal and anomalous distributions.

Figure 4: Anomaly score distribution on the synthetic data with initial $\epsilon = 0.02$, $a = 0.05$.



(a) The normal calibration set is mixed with perceived normal anomalies with a single anomaly calibration set, leading to a confounding anomaly score distribution.



(b) With the fine-grained calibration sets, we can distinguish normal and anomalous distributions.

Figure 5: Anomaly score distribution on the VitalCore data with initial $\epsilon = 0.02$, anomaly ratio $a = 0.05$.

types	*calibration*	*parallel*	*test*
2	5.1763 ± 0.4813	6.329 ± 0.5084	$7.5182 \times 10^{-6} \pm 4.4831 \times 10^{-7}$
3	5.3526 ± 0.521	5.7737 ± 0.5048	$8.9622 \times 10^{-6} \pm 5.7485 \times 10^{-7}$
4	5.4606 ± 0.4675	6.2248 ± 0.5167	$1.0615 \times 10^{-5} \pm 1.0786 \times 10^{-6}$
5	5.5443 ± 0.4664	5.9686 ± 0.5462	$1.2305 \times 10^{-5} \pm 1.6992 \times 10^{-6}$
6	5.7472 ± 0.4666	5.7497 ± 0.5147	$1.3343 \times 10^{-5} \pm 1.4019 \times 10^{-6}$
7	5.8692 ± 0.465	5.6113 ± 0.5027	$1.4792 \times 10^{-5} \pm 1.5338 \times 10^{-6}$
8	6.0652 ± 0.4685	6.1067 ± 0.5232	$1.6849 \times 10^{-5} \pm 2.3269 \times 10^{-6}$
9	6.2204 ± 0.4748	$\mathbf{5.6539 \pm 0.5335}$	$1.6838 \times 10^{-5} \pm 8.0361 \times 10^{-7}$
10	6.4894 ± 0.4868	$\mathbf{5.8449 \pm 0.5301}$	$1.9270 \times 10^{-5} \pm 2.1625 \times 10^{-6}$
11	6.6487 ± 0.5101	$\mathbf{5.8172 \pm 0.5221}$	$2.0047 \times 10^{-5} \pm 1.2425 \times 10^{-6}$

Table 6: *Calibration* and *test* time with different number of anomaly *types*. *parallel* is the time for computing the PAC thresholds with parallelization. Total calibration set size is fixed to be 10000.

between the requirements and uncertainty region. Additionally, the uncertainty is predominately affected by ϵ rather than δ. Therefore, we can shrink the uncertainty region by relaxing the ϵ, δ constraint or vice versa.

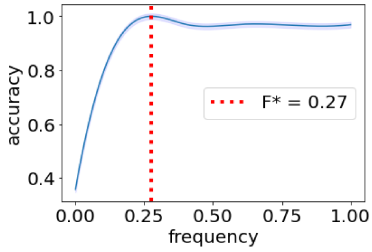


Figure 6: Update frequency and accuracy.

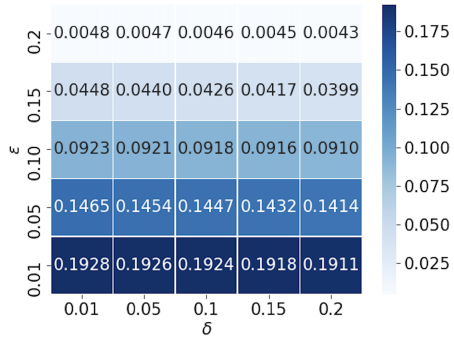


Figure 7: As ϵ and δ grow, the uncertain region shrinks.

5 DISCUSSION

One obvious concern regarding the feasibility of our framework is prompting the user for feedback. The usefulness of our framework will naturally be challenged if it needs laborious user input. However, in experiments on the VitalCore platform in real scenarios, the technicians are satisfied with providing at least one label for each anomaly type weekly. The frequency sustains the anomaly detection performance and obtains a minimum performance guarantee.

The trade-off between the user input and the tightness of the guarantee is that the PAC guarantee is asymptotic. As we obtain more data points, we are more certain about the underlying data distribution and derive a more confined guarantee. Therefore, it is up to the user to determine the level of guarantee they desire after considering the effort of providing feedback.

In addition, although we assume the normal data distribution to be relatively stable, one should continuously retrain the anomaly detector with normal data to ensure it is up-to-date. Hence, it would be beneficial if there are also feedback on normal data so as to characterize the distribution. However, we do not assume to obtain them in an online manner. If the normal data distribution shifts, the update frequency should be similar to that of the anomalies. We expect a performance enhancement when feedback on normal data becomes available.

6 CONCLUSION

We have designed a general framework to guarantee accurate performance for incremental anomaly detection in IoMT. We propose to interactively incorporate the user's judgment of evolving anomaly types to construct fine-grained anomaly calibration sets, on which we compute the PAC thresholds. We provide a two-sided guarantee on FAR and MAR based on the thresholds. Besides, our framework requires limited user input (weekly labels per class). As a side benefit, the smaller calibration size reduces the computation time, allowing for faster computation. Our framework has high accuracy and provides a theoretical guarantee for detecting evolving anomalies on synthetic and VitalCore datasets. Our method can broadly apply to ensure reliability in IoT, for example, network intrusion detection systems (IDS), industrial plants, and autonomous systems. As the next step, we look forward to conducting a user study to evaluate the level of comfort among the technicians concerning the frequency of updates. Besides, we seek to provide anomaly explanations for the predicted outcomes.

ACKNOWLEDGMENTS

This work was supported in part by NIH R01EB029767, NSF-2125561, and ARO W911NF-20-1-0080. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Institute of Health (NIH), the National Science Foundation (NSF), the Army Research Office (ARO), or the United States Government.

REFERENCES

[1] Moloud Abdar, Farhad Pourpanah, Sadiq Hussain, Dana Rezazadegan, Li Liu, Mohammad Ghavamzadeh, Paul Fieguth, Xiaochun Cao, Abbas Khosravi, U Rajendra Acharya, et al. 2021. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion* 76 (2021), 243–297.

[2] Fadi Al-Turjman, Muhammad Hassan Nawaz, and Umit Deniz Ulusar. 2020. Intelligence in the Internet of Medical Things era: A systematic review of current and future trends. *Computer Communications* 150 (2020), 644–660.

[3] Parisa Alaei and Fakhroddin Noorbehbahani. 2017. Incremental anomaly-based intrusion detection system using limited labeled data. In *2017 3th International Conference on Web Research (ICWR)*. IEEE, 178–184.

[4] Merve Astekin, Selim Özcan, and Hasan Sözer. 2019. Incremental analysis of large-scale system logs for anomaly detection. In *2019 IEEE International Conference on Big Data (Big Data)*. IEEE, 2119–2127.

[5] Monowar H Bhuyan, Dhruba K Bhattacharyya, and Jugal K Kalita. 2012. Survey on incremental approaches for network anomaly detection. *arXiv preprint arXiv:1211.4493* (2012).

[6] Elnaz Bigdeli, Mahdi Mohammadi, Bijan Raahemi, and Stan Matwin. 2018. Incremental anomaly detection using two-layer cluster-based structure. *Information Sciences* 429 (2018), 315–331.

[7] Milad Chenaghlou, Masud Moshtaghi, Christopher Leckie, and Mahsa Salehi. 2018. Online clustering for evolving data streams with online anomaly detection. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 508–521.

[8] Hyonyoung Choi, Amanda Lor, Mike Megonegal, Xiayan Ji, Amanda Watson, James Weimer, and Insup Lee. 2021. VitalCore: Analytics and Support Dashboard for Medical Device Integration. In *2021 IEEE/ACM Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE)*. 82–86. https://doi.org/10.1109/CHASE52844.2021.00016

[9] Raj Deshmukh and Inseok Hwang. 2019. Incremental-learning-based unsupervised anomaly detection algorithm for terminal airspace operations. *Journal of Aerospace Information Systems* 16, 9 (2019), 362–384.

[10] Arthur Gatouillat, Youakim Badr, Bertrand Massot, and Ervin Sejdić. 2018. Internet of medical things: A review of recent contributions dealing with cyber-physical systems in medicine. *IEEE internet of things journal* 5, 5 (2018), 3810–3822.

[11] Nico Görnitz, Marius Kloft, Konrad Rieck, and Ulf Brefeld. 2013. Toward supervised anomaly detection. *Journal of Artificial Intelligence Research* 46 (2013), 235–262.

[12] Sreelekha Guggilam, Varun Chandola, and Abani Patra. 2022. Tracking clusters and anomalies in evolving data streams. *Statistical Analysis and Data Mining: The ASA Data Science Journal* 15, 2 (2022), 156–178.

[13] John C Knight. 2002. Safety critical systems: challenges and directions. In *Proceedings of the 24th international conference on software engineering*. 547–550.

[14] Aleksandar Lazarevic, Levent Ertoz, Vipin Kumar, Aysel Ozgur, and Jaideep Srivastava. 2003. A comparative study of anomaly detection schemes in network intrusion detection. In *Proceedings of the 2003 SIAM international conference on data mining*. SIAM, 25–36.

[15] Insup Lee and Oleg Sokolsky. 2010. Medical cyber physical systems. In *Design automation conference*. IEEE, 743–748.

[16] Shuo Li, Xiayan Ji, Edgar Dobriban, Oleg Sokolsky, and Insup Lee. 2022. PAC-Wrap: Semi-Supervised PAC Anomaly Detection. https://doi.org/10.48550/ARXIV.2205.10798

[17] Yongxin Liu, Jian Wang, Jianqiang Li, Shuteng Niu, and Houbing Song. 2021. Class-incremental learning for wireless device identification in IoT. *IEEE Internet of Things Journal* 8, 23 (2021), 17227–17235.

[18] Guansong Pang, Chunhua Shen, and Anton van den Hengel. 2019. Deep anomaly detection with deviation networks. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*. 353–362.

[19] Harris Papadopoulos, Kostas Proedrou, Volodya Vovk, and Alex Gammerman. 2002. Inductive confidence machines for regression. In *European Conference on Machine Learning*. Springer, 345–356.

[20] Sangdon Park, Osbert Bastani, Nikolai Matni, and Insup Lee. 2020. PAC confidence sets for deep neural networks via calibrated prediction. *International Conference on Learning Representations (ICLR)* (2020).

[21] Sangdon Park, Edgar Dobriban, Insup Lee, and Osbert Bastani. 2022. Pac prediction sets under covariate shift. *International Conference on Learning Representations (ICLR)* (2022).

[22] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[23] Joaquin Quiñonero-Candela, Masashi Sugiyama, Anton Schwaighofer, and Neil D Lawrence. 2008. *Dataset shift in machine learning*. Mit Press.

[24] Kirthanaa Raghuraman, Monisha Senthurpandian, Monisha Shanmugasundaram, V Vaidehi, et al. 2014. Online incremental learning algorithm for anomaly detection and prediction in health care. In *2014 International Conference on Recent Trends in Information Technology*. IEEE, 1–6.

[25] Lukas Ruff, Robert A Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. 2020. Deep semi-supervised anomaly detection. *International Conference on Learning Representations (ICLR)* (2020).

[26] Kenneth Joseph Ryan and Mark Vere Culp. 2015. On semi-supervised linear regression in covariate shift problems. *The Journal of Machine Learning Research* 16, 1 (2015), 3183–3217.
[27] Mahsa Salehi and Lida Rashidi. 2018. A Survey on Anomaly detection in Evolving Data: [with Application to Forest Fire Risk Prediction]. *ACM SIGKDD Explorations Newsletter* 20, 1 (2018), 13–23.
[28] Steffen Schneider, Evgenia Rusak, Luisa Eck, Oliver Bringmann, Wieland Brendel, and Matthias Bethge. 2020. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems* 33 (2020), 11539–11551.
[29] Bernhard Schölkopf, Robert C Williamson, Alex Smola, John Shawe-Taylor, and John Platt. 1999. Support vector method for novelty detection. *Advances in neural information processing systems* 12 (1999).
[30] Sue Sendelbach and Marjorie Funk. 2013. Alarm fatigue: a patient safety concern. *AACN advanced critical care* 24, 4 (2013), 378–386.
[31] Yang Shi, Maoran Xu, Rongwen Zhao, Hao Fu, Tongshuang Wu, and Nan Cao. 2019. Interactive Context-Aware Anomaly Detection Guided by User Feedback. *IEEE Transactions on Human-Machine Systems* 49, 6 (2019), 550–559. https://doi.org/10.1109/THMS.2019.2925195
[32] Hongchao Song, Zhuqing Jiang, Aidong Men, and Bo Yang. 2017. A hybrid semi-supervised anomaly detection model for high-dimensional data. *Computational intelligence and neuroscience* 2017 (2017).
[33] Masashi Sugiyama and Motoaki Kawanabe. 2012. *Machine learning in non-stationary environments: Introduction to covariate shift adaptation.* MIT press.
[34] Masashi Sugiyama, Matthias Krauledat, and Klaus-Robert Müller. 2007. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research* 8, 5 (2007).
[35] Masashi Sugiyama, Taiji Suzuki, Shinichi Nakajima, Hisashi Kashima, Paul von Bünau, and Motoaki Kawanabe. 2008. Direct importance estimation for covariate shift adaptation. *Annals of the Institute of Statistical Mathematics* 60, 4 (2008), 699–746.
[36] Swee Chuan Tan, Kai Ming Ting, and Tony Fei Liu. 2011. Fast anomaly detection for streaming data. In *Twenty-second international joint conference on artificial intelligence.*
[37] Liang Tang, Tao Li, Florian Pinel, Larisa Shwartz, and Genady Grabarnik. 2012. Optimizing system monitoring configurations for non-actionable alerts. In *2012 IEEE Network Operations and Management Symposium.* IEEE, 34–42.
[38] Vincent Vercruyssen, Wannes Meert, Gust Verbruggen, Koen Maes, Ruben Bäumer, and Jesse Davis. 2018. Semi-Supervised Anomaly Detection with an Application to Water Analytics. In *2018 IEEE International Conference on Data Mining (ICDM).* 527–536. https://doi.org/10.1109/ICDM.2018.00068
[39] Pavithra Vijay. 2020. *Keras documentation: Timeseries anomaly detection using an Autoencoder.* https://keras.io/examples/timeseries/timeseries_anomaly_detection/
[40] Vladimir Vovk. 2012. Conditional validity of inductive conformal predictors. In *Asian conference on machine learning.* PMLR, 475–490.
[41] Vladimir Vovk, Alex Gammerman, and Glenn Shafer. 2005. *Algorithmic learning in a random world.* Springer Science & Business Media.
[42] Markus Wurzenberger, Florian Skopik, Max Landauer, Philipp Greitbauer, Roman Fiedler, and Wolfgang Kastner. 2017. Incremental clustering for semi-supervised anomaly detection applied on log data. In *Proceedings of the 12th International Conference on Availability, Reliability and Security.* 1–6.
[43] Weizun Zhao, Lishuai Li, Sameer Alam, and Yanjun Wang. 2021. An incremental clustering method for anomaly detection in flight data. *Transportation Research Part C: Emerging Technologies* 132 (2021), 103406.
[44] Aurick Zhou and Sergey Levine. 2021. Bayesian Adaptation for Covariate Shift. *Advances in Neural Information Processing Systems* 34 (2021).