Concept Drift Analysis by Dynamic Residual Projection for Effectively Detecting Botnet Cyber-Attacks in IoT Scenarios

Hanli Qiao, Boris Novikov, *Member, IEEE*, and Jan Olaf Blech, *Member, IEEE*

***Abstract*—IoT devices typically stream data such as sensor values to other devices including cloud-based services. Analyzing these streams for cyber-attacks is a challenging task. This is due to the infinite nature of stream-based datatypes. Analyzing streams can require additional real-time processing and computational performance capabilities. In this article, we focus on how concept drifts affect Botnet cyber-attack detection in IoT scenarios. To reveal the result, we incorporate the concept drift analysis to detect cyber-attacks on the Bot-IoT dataset, which consists of legitimate and simulated IoT network traffics, together with various types of attacks. We designed subdatasets of the Bot-IoT to ensure the concept drift occurs that eventually complete the experiments. The detection accuracies improved 15%–26% compared with the classification models without concept drift analysis. We also gain superior performance results by comparing confusion matrices when concept drift analysis is ongoing. We propose a technique featuring a dynamic sliding window based on the residual projection to perform concept drift analysis. During the process of finding concepts in data streams, the sample number is updated dynamically by comparing the anomalous quantity obtained by the residual projection method in the current window to the previous one. In addition to the Bot-IoT dataset, our method is also applied to two popular synthetic datasets SEA Concept and UG-2C-5D. The results demonstrate the effectiveness of our method with respect to the false alarm rate, misses, and average delay.**

***Index Terms*—Bot-IoT, concept drift, cyber-attack detection, dynamic sliding window, residual projection.**

Manuscript received February 18, 2021; revised May 16, 2021 and July 16, 2021; accepted August 20, 2021. Date of publication August 30, 2021; date of current version February 18, 2022. This work was jointly supported by Guangxi Science and Technology Base Foundation under Grant AD18281039, in part by Guangxi Natural Science Foundation under Grant 2020GXNSFBA297011, in part by the National Science Foundation of China under Grant 61862019, and in part by GLUT Scientific Research Foundation under Grant GLUTQD2017142. Paper no. TII-21-0676. *(Corresponding authors: Hanli Qiao; Jan Olaf Blech.)*

Hanli Qiao is with the College of Science, Guilin University of Technology, Center for Data Analysis and Algorithm Technology, Guilin 541004, China (e-mail: hanlinqiao77@gmail.com).

Boris Novikov is with the National Research University Higher School of Economics (HSE University), St. Petersburg 194100, Russia (e-mail: borisnov@acm.org).

Jan Olaf Blech, deceased, was with the School of Electrical Engineering, Aalto University, 02150 Espoo, Finland (e-mail: joblech@gmail.com).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2021.3108464.

Digital Object Identifier 10.1109/TII.2021.3108464

I. Introduction

INTERNET of Things (IoT) refers to a network system comprising enormous interconnected physical devices. IoT establishes information sharing and exchange among these apparatus. The concept has gained popularity in the recent decade with the advent of initiatives such as Industry 4.0. However, these devices are easily attacked and are considered vulnerable points in the IoT network. Therefore, the study of IoT cybersecurity is quite important and notable.

In an IoT system, an intrusion detection system (IDS) is a security mechanism that monitors network traffic particularly in the network layer. However, the IDS in an IoT setup faces more challenges than traditional networks. Its inherent nature, such as bandwidth restrictions, power limitations, and various connection possibilities, cause nontrivial challenges differing from conventional networking environments. The IDS design should take possible tradeoffs between accuracy and performance into account [1], [2].

The two types of solutions for IDS are signature-based and anomaly based [3]. Signature-based IDS monitor network packets and compare them to the *signatures,* which are the predetermined attack patterns. Although this approach could detect known attacks easily, it is difficult to detect new attacks. Anomaly based solutions perform better in the latter case. Anomalous behaviors exist ubiquitously in the real world. Designing feasible and appropriate approaches to detect such unexpected situations is quite practicable and promising for applications from various areas. A variety of techniques have been applied to anomaly detection in diverse fields successfully. However, many of them focus on static samples, and run counter to the actual situation, where data are generated continuously such as in streams. The demand for a stream-based solution is even more pressing due to the rapid developments of Big Data, IoT, and artificial intelligence. The typical property of a stream is that data arrives continuously and tends to change over time. This property leads to challenges of infinity, nonstationary, memory limitation, and real-time analysis when preprocessing IoT data streams. In addition to recent research on secure IIoT [4], intrusion detection for IoT networks [5], and cyber-attacks on generator synchronization processes [6], some more detailed analysis scenarios of IoT data streams comprise network monitoring [7], [8] and predictive maintenance [9], [10]. Most of these applications rely on the techniques of data stream mining, i.e., clustering and classification.

We mentioned beforehand that data in streams arrives continuously and tend to change over time. Generally, we define a concept as the data distribution for a sliding window in a stream. In the real world, concepts in a data stream often change with time rather than staying stable. This means that in data streams, the concepts of interest depend on unknown context and may alter in unforeseen ways. From the view of machine learning and predictive analytics, this phenomenon generally is known as concept drift. Concept drifts may affect the precision of data analysis. For example, the predictions can become less accurate over time. An intuitive example of concept drift are the behavior patterns of online shopping. The purchasing preferences of customers may change over time. For example, a model forecasting someone's buying preferences may become increasingly less accurate over time—this is a concept drift. In this case, potential reasons for concept drift may be relevant to product characteristics, merchant reputation, services, regions, or special periods. Goods like gifts are selected more frequently than usual in special occasions such as festivals, and there is less will to purchase something if the merchant reputation is low. However, the drifts often occur in unforeseen ways, which means that we may do not clearly see when, where, and why the concepts have changed. Therefore, the analysis of concept drift is essential for various tasks of IoT data stream mining. It is especially true in the case of cyber-attack detection because a concept drift in the normal network traffic behavior might wrongly trigger an alarm.

By means of the differences between newer concepts and older concepts, concept drift usually can be classified into abrupt (also referred to as *sudden*), gradual, incremental, and reoccurring [11], [12]. Diverse application tasks for which the problems of concept drift are specifically relevant are categorized in [11]. The authors state that the primary tasks related to concept drift are prediction, classification, detection, clustering, and itemset mining.

Our main task in this article is to study cyber-attack detection for IDS incorporating concept drift analysis. To avoid ambiguity, we use the phase cyber-attack detection to represent the anomaly detection in IDS. We aim to study how concept drift exiting in IoT scenarios affects its task of cyber-attack detection. An effective detection strategy will provide reasonable alarms on time when an IoT network system is under attacks.

To meet our goal, we divide our research into two parts. First, we build a model and an algorithm to detect the occurrence of concept drift on two popular synthetic datasets: SEA Concept and UG-2C-5D. Second, our concept drift analysis based on the provided model is evaluated with respect to the performance of cyber-attack detection on a realistic Botnet dataset in IoT (abbreviate as Bot-IoT) [13]. In the phase of concept drift detection, three metrics are exploited to access the model. They are false alarm (FP), *missing* (FN), and average delay. FP stands for the wrongly detected instances of drift points. *Missing* stands for the drift points that are not reported. In the phase of cyber-attack detection, the detailed descriptions of the metrics are introduced in Sections IV-B and IV-C.

In this article, we propose a dynamic sliding window method based on residual subspace projection to study how concept drift analysis affects the performance of cyber-attack detection in IoT scenarios. Our main contributions are summarized as follows:

1) The proposed method for concept drift detection is data-based, it does not relay on information about labels or statistics.
2) Based on the technique of residual subspace projection, we transform the detection of concept drift to calculate the anomalous quantity with the help of a dynamic sliding windows. This is a simple, cheap, and effective approach.
3) We provide a novel idea to detect cyber-attacks in IoT scenarios, which is considering the influence of concept drift.

II. Related Works

In recent years, attacks against cyber-physical systems have been increasing. For this reason, research on cybersecurity in industrial control systems has been conducted. For example, a cyber-attack detection system based on four classical classification models is described in [14]. Similarly, in [15] the authors provide a strategy integrating nonlinear control and neural networks for nonlinear CPS. A specific concern about false-data injection attack detection is studied in [16]. It is clear that most of these provided approaches for cyber-attack detection in IoT system make use of machine learning-based techniques and show their promising performances in diverse domains.

Cyber-attack detection plays a significant role in IoT. Once intruded, attackers can use IoT devices to collect information from the network or launch attacks on other systems. The study of [17] states that cyber-attacks in the level of insecure interfaces is a high-level security issue, which is mainly related to the applications executing on IoT. The authors in [18] provided a comprehensive study about deep learning for cybersecurity intrusion detection. They list various deep learning techniques and widely used datasets. Among these surveyed datasets, only the Bot-IoT is IoT traffic based. This in numerous relevant research works, such as the application in an IIoT scenario of smart cities by a two-level deep learning framework in Botnet detection system [19]. Similarly, researchers in [20] proposed a new framework and a hybrid algorithm to select an effective machine learning technique for cyber-attack detection for smart cities. The verification dataset is Bot-IoT. To avoid inappropriate feature selection in machine learning based techniques, the authors developed a novel metric "CorrAUC" to design a new feature selection algorithm on the Bot-IoT dataset in [21]. Deep learning skills are the same popular choices in the relevant applications from literatures [22], [23]. They focus on network forensic framework and smart grids, respectively.

Apparently, deep learning is invaluable in detecting IoT-based attacks. However, most studies concentrate on IoT security from the views of applications, algorithms, and feature selection. However, few of them consider the concept drift, which is a common phenomenon in IoT scenarios. Therefore, in this article, we aim to reveal how the analysis of concept drift affects Botnet cyber-attack detection in IoT. To satisfy this purpose, we compare the performances with concept drift analysis to ones

without concept drift analysis. We design experiments on the Bot-IoT. This dataset was released in Nov 2018 [13] and is available at the website.[1] More details about the Bot-IoT will be described in Section IV-B.

Detection of botnet attacks in IoT scenarios is crucial for cybersecurity. To get efficient implementation in memory-constrained IoT devices, the authors proposed a hybrid deep learning method LAE-BLSTM (long short-term memory autoencoder-bidirectional long short-term memory) [24]. LAE is responsible for dimensionality reduction, while BLSTM contributes to classification on the same dataset Bot-IoT. We will differently consider the dynamic property in IoT botnet attack detection. That is incorporating concept drift analysis with deep classifiers to power their capabilities. Another similar work put forward an edge-centric IoT defense scheme FlowGuard against distributed denial-of-service (DDoS) attacks [25]. The two components of flow filter and flow handler apart play the roles of filtering malicious flows and DDoS identification. Convolution neural network (CNN) and LSTM in charge of DDoS classification with superior performances. However, the authors only consider the specific botnet attack DDoS. In this article, we leverage CNN and LSTM to comprehensively detect the botnet attacks of DoS, DDoS, and Reconnaissance.

During analyzing concept drift, we develop a novel model to detect drift points. Other techniques such as density difference estimation based [26] and stochastic approach [27] are provided for detecting concept drift, respectively. Generally, concept drift detection strategies are classified into active and passive approaches [28]. Active algorithms are further classified into methods that are based on label detection and those that are not. However, in the real-world, the data labels are usually unavailable. Therefore, to detect concept drifts, we build a method that does not rely on labels.

III. Proposed Approach

In this article, the problems we mainly focus on are concept drift analysis and cyber-attack detection. First, we briefly introduce the problem statement in the forthcoming section.

A. Problem Statement

For concept drift analysis, we assume that the streaming data is generated by a function f. We define concept drift as f changing over time under unforeseen ways. The main problem we want to examine is whether and when f changes. Furthermore, we investigate how this can serve network attack detection. Unlike other strategies, for detecting concept drift, the model we provide does not require the label information. This fits the real world fact that the complete streams are usually not available and the lack of labels.

For attack detection with concept drift analysis, the main steps are first to detect whether the concept drifts happen and then to design an efficient method to detect attacks on each concept.

To finish the tasks of concept drift analysis and attack detection, we design a dynamic sliding window method. This method is based on the residual subspace projection technique, which was originally proposed for control system theory in the 20th century. In [29], the authors apply the residual subspace analysis in anomaly detection to a large-scale data stream network. Our approach relies on a reasonable assumption. According to our definition about concept drift. We assume f_{t_1} is the observed data function at time t_1. After sometime, it changes to f'_{t_2} at t_2. Then the arriving data $f'_{t_2}(\mathbf{x})$ is more likely considered to be abnormal for the underlying concept of f_{t_1} if f' and f are significantly different. This means if the data underlies a different concept, the anomalous quantity will increase.

B. Residual Subspace Projection

Concept drift detection methods can benefit from the help of data labels, such as error classification rate methods. However, in the real application, the label of data streams is often unavailable. Unlike what happens in other domains, it is not only expensive but also very hard to gather and label data to train classifiers. This is because real attack data is not abundant and often complex to label. For overcoming this limit and discovering a more reasonable method, we investigate how to satisfy the purpose of concept drift detection that does not use data labels. We proceed from the data features to carry out our research. Based on our assumption that when a concept drifts, the anomalous instances should increase rapidly. Therefore, investigating how to compute the anomalous quantity is essential. In this article, we exploit the residual subspace projection method to capture the increased anomaly quantity.

Given a sample matrix $\mathcal{S} = \{s_1; \cdots ; s_n\}$, where each row represents a sample instance $s_i \in \mathbb{R}^D$. D is the data dimension. For simplicity, we assume $\mathcal{S}$ has been centralized. The eigenvalue decomposition of the covariance matrix $\mathcal{S}$ is

$$\Sigma_{\mathcal{S}} = \frac{1}{n}\mathcal{S}^T\mathcal{S} = \mathbf{U}\Lambda\mathbf{U}^T. \tag{1}$$

where $\mathbf{U}$ is composed of the eigenvectors corresponding to the selected eigenvalues, representing principal components. The residual subspace projection is given by

$$proj\_residual = \mathcal{S}(\mathbf{I} - \mathbf{U}\mathbf{U}^{\mathbf{T}}). \tag{2}$$

In residual subspace analysis, *proj_residual* is assumed to be multivariate normally distributed. Based on this assumption, the squared prediction error (SPE) $\|proj\_residual\|_2^2$ follows a noncentral Chi-square distribution. The null hypothesis is stating that the data is normal. Therefore, it is possible to base the rejection of the null hypothesis on the norm of the error vector. It is rejected if the norm exceeds a certain threshold. This threshold should correspond to the desired confidence value. The threshold is Q-statistic, which is a function of nonprincipal eigenvalues in the residual subspace. In order to obtain the Q-statistic, we need to provide a significance level α. Then, the Q-statistic can be calculated in the following way:

$$Q_\alpha = \theta_1 \left[\frac{c_\alpha \sqrt{2\theta_2 h_0^2}}{\theta_1} + 1 + \frac{\theta_2 h_0 (h_0 - 1)}{\theta_1^2} \right]^{\frac{1}{h_0}} \tag{3}$$

where $h_0 = 1 - \frac{2\theta_1\theta_3}{3\theta_2^2}$, $\theta_i = \sum_{p+1}^{D} \lambda_j^i,\ i = 1, 2, 3$. $C_\alpha = 1 - \alpha$ is the confidence level. α is the desired false alarm rate, and

[1][Online]. Available: https://cloudstor.aarnet.edu.au/plus/s/lfTuAPJmQTaKs2s

$\lambda_j,\ j = 1, \ldots, D$ are the eigenvalues of $\Sigma_{\mathcal{S}}$. An anomaly is detected whenever $\|proj\_residual\|_2^2 > Q_\alpha$, otherwise it is considered as normal.

C. Dynamic Residual Projection Method

We work with the assumption, that once the anomaly quantity in the current concept is larger than a threshold, we have discovered a concept drift. For completing this purpose, we use a dynamic sliding window based on the technique of residual subspace projection to capture anomaly samples. We abbreviate the method as DRPM.

DRPM uses two types of windows. With respect to size, one is dynamic and another one is fixed. The dynamic window is called *concept window* with capacity σ. We denote it as W_1, whereas the fixed window is W_2 with size γ. The relationship between them is $\gamma < \sigma$. The pseudocode in Algorithm 1 states the implementation of DRPM. Where *step* is the sliding step and p is the number of principal components composed to $\mathbf{U}$.

The initial window W_1 contains the initial γ samples, we compute and store its Q_α, residual subspace $\tilde{\mathbf{U}} = \mathbf{I} - \mathbf{U}\mathbf{U}^{\mathbf{T}}$, and anomaly quantity Q_1. We set the sliding step as *step* to obtain W_2, and then project all its samples to $\tilde{\mathbf{U}}$. The next step is to apply $Q-$statistic to compute the anomaly quantity Q_2. When Q_2 is larger Q_1 than a threshold, the drift is detected and $W_1 = W_2$. Then, we update $Q_\alpha,\ \tilde{\mathbf{U}}$, and Q_1. If $Q_2 \leq Q_1$, or $Q_2 - Q_1$ is smaller than a threshold, then $W_1 = [W_1 \bullet W_2]$, which means putting W_2 into W_1 and removing the duplicate samples. We repeat the above steps until the size of W_1 arrives at σ. Then we update again $Q_\alpha,\ \tilde{\mathbf{U}}$, and Q_1. Now, W_2 should start with the next sample immediately following the last sample of W_1. We repeat the cycle until all samples scanned.

Algorithm 1: Dynamic Residual Projection Method-DRPM.

Input: $\sigma,\ \gamma,\ step,\ p,\ data$
Output: $concept$

for k=1:s **do**
  if k==1 **then**
    $W_1 = data[(k-1)step + 1 : (k-1)step + \gamma, :]$;
    compute $\mathbf{U}$ **and** Q_α **of** W_1;
    $proj\_residual = W_1(\mathbf{I} - \mathbf{U}\mathbf{U}^{\mathbf{T}})$;
    for j=1:σ **do**
      $C(j,k) = \text{norm}(proj\_residual(j,:))$;
      $Put\ the\ anomalous\ quantity\ into\ D(k)$;
  else
    $W_2 = data[(k-1+a)step+1 : (k-1+a)step+\gamma, :]$;
    $proj\_residual = W_2(\mathbf{I} - \mathbf{U}\mathbf{U}^{\mathbf{T}})$;
    for j=1:σ **do**
      $C(j,k) = \text{norm}(proj\_residual(j,:))$;
      $Put\ the\ anomalous\ quantity\ into\ D(k)$;
    if $D(k) - D(k-1) > threshold$ **then**
      $W_1 = W_2$; update $\mathbf{U},\ Q_\alpha,\ C,$ and $D(k)$;
      if $a > 0$ **then**
        $concept(k,1) = k$;
        $concept(k,2) = (k-1+a)step + 1$;
    else
      $W_1 = [W_1 \bullet W_2]$;
    if *the size of* W_1 *arrives* σ **then**
      update $\mathbf{U},\ \mathbf{Q}_\alpha$;
      $a = a + 1$;

IV. Experiments and Analysis

We present our experimental results on concept drift analysis for cyber-attack detection. The concept drift detection experiments are carried out on the datasets SEA Concept and UG-2C-5D to assess the performance in section IV-A. Furthermore, we apply DRPM on a reconstructed subdataset containing abrupt concept drifts of the Bot-IoT in Section IV-B. How their results serve in IoT cyber-attack detection will detail interpret in Section IV-C.

A. Concept Drift Detection

We perform the dynamic sliding window method onto SEA Concept and UG-2C-5D. In this task, the metrics are TP, FP, FN, and the average delay (abbreviated as Delay in this article). Delay can be obtained by the following formula:

$$Delay = \frac{\Sigma delay\ at\ each\ concept\ drift}{the\ total\ number\ of\ detected\ drifts}.$$

First, we state the distributions of SEA Concept and UG-2C-5D in this paragraph. The dataset SEA concept can download from the website.[2] There are 60 000 samples with three attributes and two classes. Attributes are numeric values between 0 and 10, however, only the first two are relevant. In SEA concept, there are four concepts with three drift points, 15 000 examples each. The dataset has around 10% noise. Similarly, UG_2C_5D is collected from [30], which contains 200 000 samples with five attributes. There are 100 concepts in total and drifts occur at every 2000th sample. Therefore, we have 99 drift points.

Figs. 1 and 2 show the results of detected drifts on SEA concept and UG-2C-5D, respectively. In both figures, the * are the detected drift points. The x-axis is the detection window, which computes the anomalous number, and the y-axis is the anomalous quantity at each detected window. The filled • represents window W_1 of size δ. Differing from Fig. 2, the red dash lines in Fig. 1 represent the time when the real concept drift occurs. The subfigure (a) in Fig. 1 shows the results in the case of $\sigma = 2500, \gamma = 500,\ step = 250$. We detected four drift points with the corresponding metrics, where FP=1, FN=0, and

$$Delay = \frac{1500 + 2250 + 1500}{3} = 1750.$$

We change the capacity of window W_1 to 2000 in the subfigure (b) and keep the other parameters the same. In this case, we have three wrongly detected drift points, furthermore

$$Delay = \frac{3750 + 2250 + 10500}{3} = 5500$$

is the same as (a) where there are no missing drift points.

To evaluate the effect of different parameters in concept drift detection on UG-2C-5D, we design the experiments with two aspects in mind. They are window size and *step*. Fig. 2 displays results matching our intuition. In detail, the figure displays the

[2][Online]. Available: https://github.com/leslin1010/SeaConceptDataset

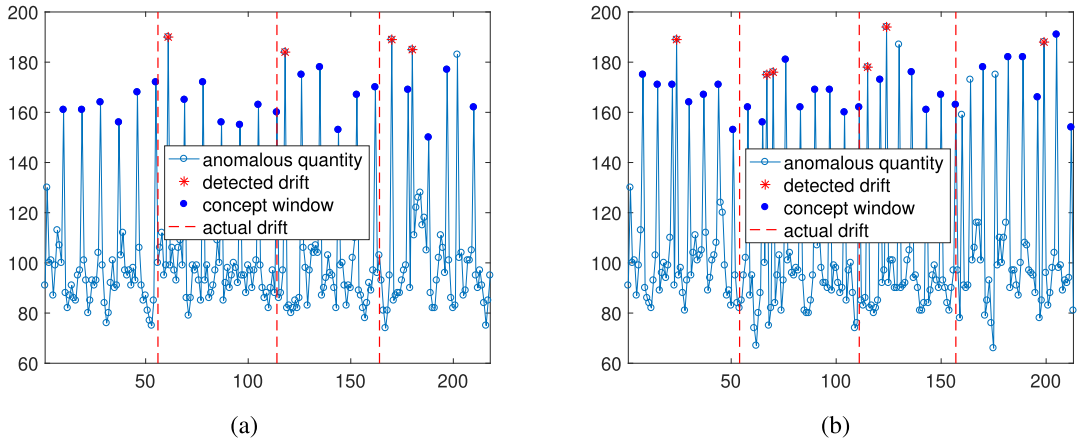


Fig. 1. Illustrations of detected drift on SEA concept with $\alpha = 0.001,\ p = 2$. (a) $\sigma = 2500, \gamma = 500,\ step = 250$. (b) $\sigma = 2000, \gamma = 500,\ step = 250$.

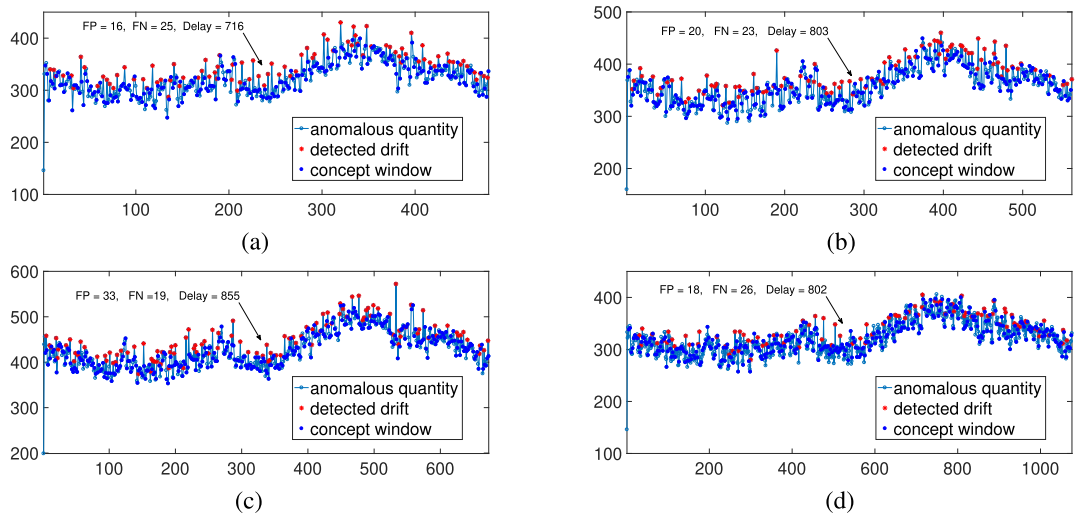


Fig. 2. Illustrations of detected drift on UG-2C-5D with $\alpha = 0.001,\ p = 3$. (a) $\sigma = 1500,\ \gamma = 900,\ step = 280$. (b) $\sigma = 1500,\ \gamma = 1000,\ step = 240$. (c) $\sigma = 1500,\ \gamma = 1200,\ step = 200$. (d) $\sigma = 2000,\ \gamma = 900,\ step = 140$.

detected drifts on UG-2C-5D with different parameters. The window W_1 of the first three subfigures has the same size $\sigma = 1500$. The size of window W_2 and $step$ are different. The parameters of subfigure (d) are $\sigma = 2000, \gamma = 900,\ step = 140$.

1) Variables Determination by Hoeffding Bound: The main variables in DRPM are σ, γ, and $step$. We gain the above results by experimental values. However, sliding window-based techniques commonly sensitive to these variables. We update to provide a reasonable rule to choose their appropriate values by the Hoeffding bound.

For a real-valued random variable r whose range is R, suppose we have n independent observations of r. Hoeffding bound illustrates that with confidence $1 - \delta$, the true mean is at least $\bar{\mu} - \epsilon$, $\bar{\mu}$ is the mean of these n observations, where

$$\epsilon = \sqrt{\frac{R^2 \ln(1/\delta)}{2n}} \tag{4}$$

here the range R is 1 for a probability, whereas $R = \log_2 C$ (C is the class number) for an information gain [31]. In the case of SEA concept and UG-2C-5D, $R = 1$.

With the help of the Hoeffding bound shown in (4), we could acquire the minimum of σ under specific conditions. To reserve the confidence consistent in DRPM, we default to set it as 0.999, i.e., $\delta = 10^{-3}$. ϵ commonly is 0.05 [32]. Due to the property of monotonically decreasing, we calculate $\sigma_{\min} = \lceil \frac{R^2 \ln(1/\delta)}{2\epsilon^2} \rceil = 1382$. The rules we design for γ and $step$ are $\gamma = \lfloor \frac{1}{5}\sigma \rfloor$ and $step = \lfloor \frac{1}{4}\gamma \rfloor$.

Table I exhibits the results of drift detection on SEA concept and UG-2C-5D using DRPM, whose main variables are calculated based on the Hoeffding bound. We provide a feasible start to compute these variables. The Hoeffding bound-based approach may lead to $missing$ increases due to minimized values used.

B. Bot-IoT Dataset and Its Concept Drift Detection

IoT-based botnet attack is one of the most critical threats on the internet. They spread faster and perform more impacts than other attacks [33], [34]. Our mission is to study how the analysis of concept drift affects botnet cyber-attack detection

TABLE I
RESULTS ON SEA CONCEPT AND UG-2C-5D USING DRPM WHOSE VARIABLES BASED ON THE HOEFFDING BOUND

$\sigma = 1382, \gamma = 276,\ step = 69$	SEA Concept	UG-2C-5D
TP	2	62
FP	0	16
FN	1	37
$Delay$	3720	918

TABLE II
DESIGNED CONCEPTS OF BOT-IOT

	Normal	DDoS	Reconnaissance	DoS
C_1	2000	2000	10000	2000
C_2	2000	10000	2000	2000
C_3	2000	2000	2000	10000
C_4	2363	2000	2000	2000
C_5	0	4000	4000	4000

TABLE III
FEATURE DESCRIPTIONS OF BOT-IOT

Feature	Description
seq	Argus sequence number-Argus tool was used for extracting features
stddev	Standard deviation of aggregated records
min	Minimum duration of aggregated records
state_number	Numerical representation of feature state
mean	Average duration of aggregated records
drate	Destination-to-source packets per second
srate	Source-to-destination packets per second
max	Maximum duration of aggregated records

in IoT scenarios. Specifically, we leverage the Bot-IoT to meet this goal. Compared to other well-known datasets widely used in IDS or network forensic systems, the Bot-IoT is superior in possessing all required features, such as realistic traffic, labeled data, and IoT traces.

To simulate the network behaviors of different IoT devices, the authors employed the Node-red tool that is a visual open-source tool for wiring IoT. Using this tool, they mimicked various IoT sensors, such as temperature, pressure, and humidity. Bot-IoT features an IoT scenario, where a weather station is connected to its testbed to handle air information of pressure, humidity, and temperature. Furthermore, they applied the following IoT scenarios, a smart fridge, motion-activated lights, a remotely activated garage-door, and a thermostat. This configuration is often used in the context of *smarthome*. The five scenarios connect to both the Ubuntu server and the AWS IoT hub to simulate regular IoT traffic.

An intrusion detection and prevention system (IDPS) should be successful when countering a feasible and sensible threat model built in the early stage during the design phase. In the case of *smarthome*, the authors leveraged STRIDE and VAST threat modeling approach to identify and mitigate the botnet attacks in [35]. The proposed threat model follows up the steps comprised of use case description, security requirements analysis, data- and process-flow diagram generation, threats and botnet identification, and mitigation. They provide the development- and application-level threats identification. Beyond a feasible threat model, an accurate and effective botnet detection technique is essential in IDPS. However, the concept drift often impacts cyber-attack detection under a nonstationary environment.

There are over 70 million traffics of normal and attacks collected in Bot-IoT. The main categories of cyber-attacks are probing attacks, denial of service (DoS), and information theft. To investigate how concept drifts effects cyber-attack detection in IoT Botnet scenarios, our primary work is to preprocess the Bot-IoT to ensure the concept drift exists. The work is done based on its changed distribution at each concept. To get a reasonable subdataset from the Bot-IoT, we only consider the attacks which have a large amount. They are reconnaissance, DDoS, and DoS. We randomly select 8363 normal for benign scenarios and 20 000 attacks for each type of Botnet. There are 68 363 instances in the subdataset, to keep it simple, we still denote it by Bot-IoT in the rest of this article. We eventually designed five concepts based on diverse distributions, as shown in Table II. The former three concepts represent the phenomenon of massive specific attacks that suddenly appeared in a period. The fourth concept indicates a common condition in which the number of normal and attacks is similar. The last concept shows that in a period, the three attacks emerge at the same frequency. To provide intuitive interpretation, in Fig. 3, we apply the same percent as each distribution to mimic the different concepts of Bot-IoT in Table II.

For each concept, all instances are randomly sorted to provide more reasonable results. According to [13], we employ the 10 best features, *seq*, *stddev*, *N_IN_Conn_P_SrcIP*, *min*, *state_number*, *mean*, *N_IN_Conn_P_DstIP*, *drate*, *srate*, *max*. Where *N_IN_Conn_P_SrcIP* and *N_IN_Conn_P_DstIP* are the number of inbound connections per source or destination IP. We reacquire the two features of Bot-IoT by accounting its number for each source or destination IP per 100 instances on the designed Bot-IoT. We display descriptions of the rest eight features in Table III.

From Table II, we could easily find that the real drift points on Bot-IoT are 16001, 32001, 48001, and 56364. We have four classes on Bot-IoT, therefore $R = 2$. Owing to the complexity, we enlarger ϵ as 0.07, then based on the Hoeffding bound, we calculate the variables are $\sigma = 2820,\ \gamma = 564,\ step = 141$. Similarly, there are 2 $missing$ drift points. We start from these values to choose more reasonable variables to get improved conclusions. Table IV shows the drift detection results when various variables chosen by DRPM. The bold numbers are correctly detected drift points, however, with delay, whereas the regular ones are wrongly reported drifts. When $\sigma = 3000$, $\gamma = 1000,\ step = 250, p = 6$, one drift point is missing and its FP and $Delay$ is smallest among the three conditions. In Section IV-C, we will discuss how these results serve in cyber-attack detection of Bot-IoT in detail.

C. Effects of Concept Drift Analysis in Detecting Cyber-Attacks

For mitigating the risks of concept drifts in data streams, an effective detection mechanism featuring an adaptive learner can be used. Once the drift is detected, the learner will adaptively learn the characteristics in a novel concept. Generally, there are several ways to address concept drift: model retraining, feature dropping, and ensemble learning are the most popular ones. We adopt model retraining to handle concept drift in this article. The

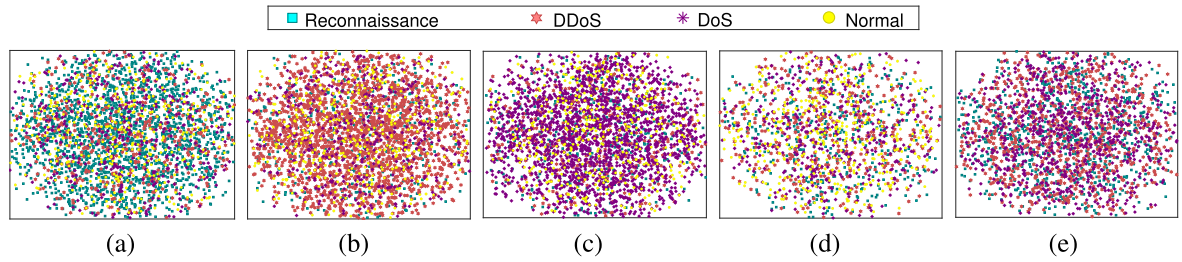


Fig. 3. Mimic distribution diagrams in different concepts of Bot-IoT. (a) First concept. (b) Second concept. (c) Third concept. (d) Fourth concept. (e) Fifth concept.

TABLE IV
RESULTS OF DRIFT POINT DETECTION ON BOT-IOT BY DRPM WITH $\alpha = 0.001$

Evaluation metrics														FP	FN	*Delay*
Real drifts		-	-	-	**16001**	-	-	**32001**	-	-	**48001**	**56364**	-	0	0	0
DRPM drifts	$\sigma = 3000,\ \gamma = 1000,\ step = 500,\ p = 4$	6001	13501	15501	**17001**	-	-	**39001**	-	-	**48001**	**62001**	65001	4	0	3409
	$\sigma = 4500,\ \gamma = 1500,\ step = 750,\ p = 7$	15001	-	-	**16501**	24001	29251	**47251**	-	-	**48001**	**60751**	-	3	0	5034
	$\sigma = 3000,\ \gamma = 1000,\ step = 250,\ p = 6$	13501	-	-	-	-	-	**38751**	47251	47501	**48001**	**57001**	-	3	1	2462

TABLE V
ACCURACY ON BOT-IOT OBTAINED BY CNN

DRPM reported concepts	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	Average
$\sigma = 3000,\ \gamma = 1000,\ step = 500$	98.33%	98.21%	95.86%	92.76%	88.78%	97.97%	97.61%	98.12%	97.41%	**96.12%**
$\sigma = 4500,\ \gamma = 1000,\ step = 750$	99.10%	85.14%	95.85%	93.74%	98.51%	85.93%	97.47%	99.40%	-	94.39%
$\sigma = 3000,\ \gamma = 1000,\ step = 250$	99.11%	49.30%	97.85%	76.14%	75.43%	98.73%	99.04%	-	-	85.09%
exactly concepts	99.20%	94.95%	98.61%	99.42%	98.98%	-	-	-	-	**98.23%**
without concept drift analysis	99.20%	74.08%	34.67%	66.61%	76.98%	-	-	-	-	70.31%

TABLE VI
ACCURACY ON BOT-IOT OBTAINED BY LSTM

DRPM reported concepts	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	Average
$\sigma = 3000,\ \gamma = 1000,\ step = 500$	97.38%	97.33%	93.29%	83.05%	85.64%	95.84%	95.51%	96.33%	96.61%	**93.44%**
$\sigma = 4500,\ \gamma = 1000,\ step = 750$	97.62%	77.71%	95.43%	93.47%	95.89%	74.52%	94.22%	95.16%	-	90.50%
$\sigma = 3000,\ \gamma = 1000,\ step = 250$	97.95%	94.68%	95.56%	78.41%	75.43%	98.63%	96.23%	-	-	90.98%
exactly concepts	98.14%	94.57%	96.86%	99.18%	96.57%	-	-	-	-	**97.06%**
without concept drift analysis	96.25%	83.90%	33.54%	52.30%	74.52%	-	-	-	-	68.10%

main idea is once a drift point detected on Bot-IoT, we retrain a classifier immediately. In all experiments, we use 65% instances to train and the rest of 35% to validate.

In solving complicated problems like cybersecurity in the IoT world, deep learning is a promising technique [36]. CNN and LSTM are popular choices and are successfully applied in cyber security-related areas [37]. We thus employ CNN and LSTM to compare the performances of cyber-attack detection on Bot-IoT with and without concept drift analysis.

1) CNN: is a standard deep neural network technique that generally includes convolutional, pooling, and fully connected layers. CNN is successfully applied in image classification originally. It has been used for other machine learning tasks [38].
2) LSTM: is a special kind of recurrent neural networks (RNNs). LSTM units could improve the *vanishing gradient* problem that may encounter when training traditional RNNs [39]. Generally, LSTM units contain a memory cell and three gates, i.e., an input gate, an output gate, and a forget gate.

In this article, CNN and LSTM are responsible for completing the cyber-attack detection problem on Bot-IoT. In fact, this is a multiclass classification task. We are interested in the effects of concept drift analysis by DRPM for detecting cyber-attacks on Bot-IoT. Tables V and VI reveal the detection accuracy of with and without concept drift analysis obtained by CNN and LSTM. The accuracy is the percentage of correctly detected samples in each concept. We can observe that if we never update classifiers, the performance becomes worse. In the case of CNN, the detection accuracy drops down to 34.67% in the third concept from initial 99.20%. Similarly, in the third and fourth concepts, the accuracies obtained by LSTM fall to 33.54% and 52.30%. Oppositely, if we could find exactly drift points, the accuracy average raises to 98.23% by CNN and 97.06% by LSTM.

However, it is hard to find the exact drift points. Generally, there exist false reports (FP), missing (FN), and delay ($Delay$), as Table IV shows. From Tables V and VI, we find that FN and $Delay$ play a more vital role compared to FP. Especially for CNN, FN is much more dangerous since it relies heavily on training data. When $\sigma = 3000,\ \gamma = 1000,\ step = 250$, DRPM

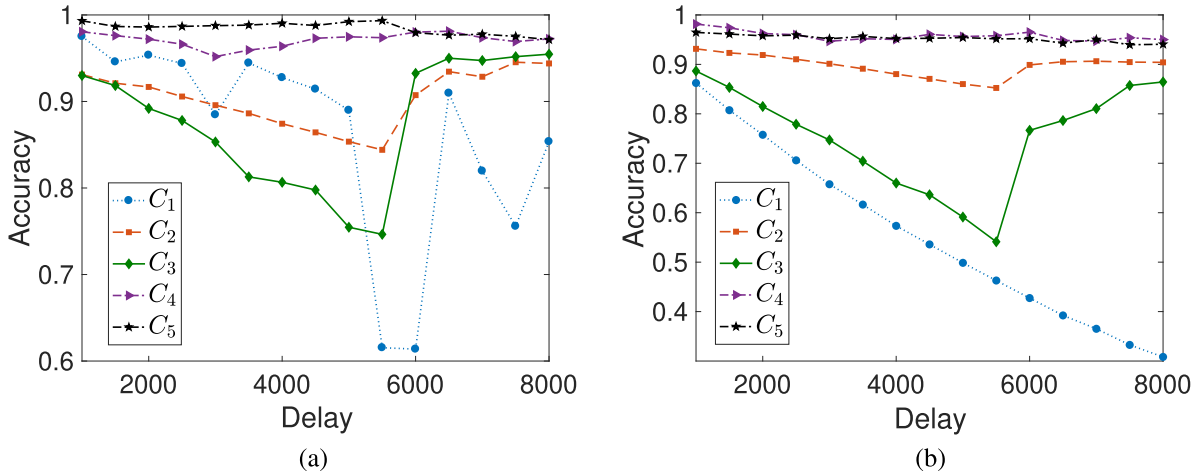


Fig. 4. Accuracy under various $Delay$ by CNN and LSTM on Bot-IoT. (a) CNN classifier. (b) LSTM classifier.

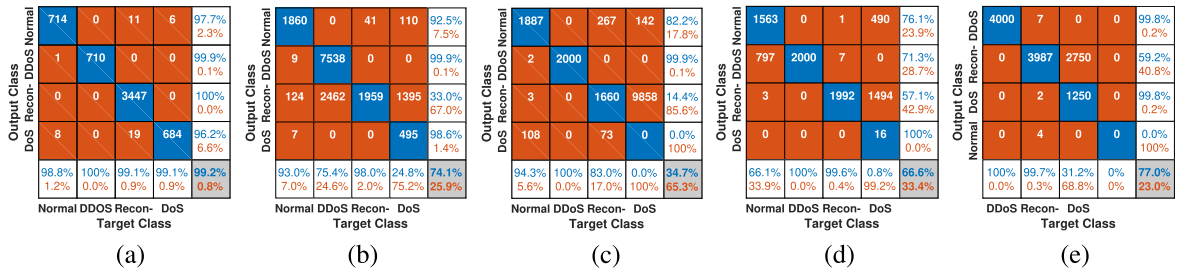


Fig. 5. Results without concept drift analysis obtained by CNN on Bot-IoT. (a) First concept. (b) Second concept. (c) Third concept. (d) Fourth concept. (e) Fifth concept.

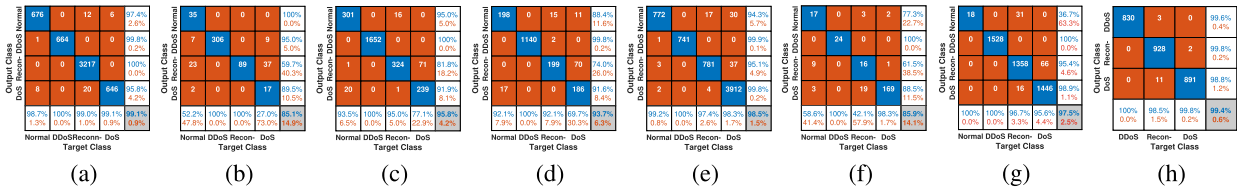


Fig. 6. Confusion matrices obtained by CNN when $\delta = 4500,\ \gamma = 1000,\ step = 750$. (a) C_1. (b) C_2. (c) C_3. (d) C_4. (e) C_5. (f) C_6. (g) C_7. (h) C_8.

lost a drift point, which leads to accuracy in the second concept declining to 49.30% from 99.11%. Under the same condition, LSTM performs much better than CNN maybe due to its memory property. When $\sigma = 4500,\ \gamma = 1000,\ step = 750$, $Delay$ is the largest, accuracy average both in CNN and LSTM fell down.

Fig. 4 shows the accuracy of CNN and LSTM under various delays on Bot-IoT. We suppose that FP-, FN-free, and the delay of each drift point are the same. $Delay$ starts at 1000 and increases every 500 to the end of 8000. Fig. 4 illustrates that $Delay$ affects the first three concepts of Bot-IoT more severely. The larger $Delay$ is, the worse performances they contribute until $Delay$ is 5500. For example, the accuracy of C_3 using CNN is 94.48% when $Delay$ is 1000, whereas 77.07% when $Delay$ is 5500. The accuracy of C_1 using LSTM declines from 86.24% to 30.83%.

However, some exceptions exist. At some special concepts, $Delay$ is large enough the accuracy rises reversely like C_2 and C_3. Additionally, the distributions of C_4 and C_5 are relatively balanced to be safe to face larger Delay. Yet, in the real IoT world, the network streams usually pretty complex with plenty of diverse concepts. Thereby larger delay in concept drift analysis is dangerous for cyber-attack detection in IoT scenarios under a nonstationary environment.

Although the performance will be affected under different conditions, the cyber-attack detection ability raises 15%–26% percent after the concept drifts analysis on Bot-IoT.

Especially in imbalanced datasets, accuracy is not an ideal evaluation metric. To avoid this bias, we show confusion matrices to provide intuitive comparisons in Figs. 5–7. Where "Recon-" represents reconnaissance, "target class" is real while "output class" is detected. It is clear that if we train a

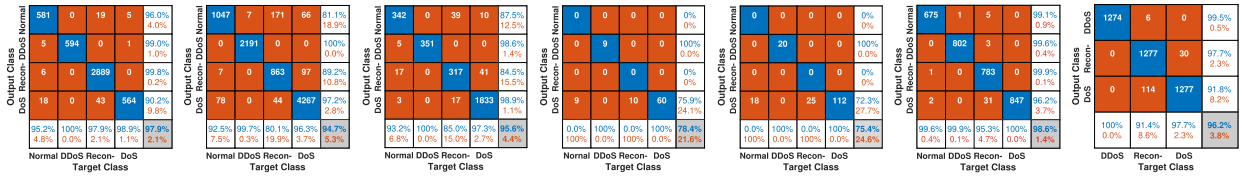


Fig. 7. Confusion matrices obtained by LSTM when $\delta = 3000$, $\gamma = 1000$, $step = 250$. (a) C_1. (b) C_2. (c) C_3. (d) C_4. (e) C_5. (f) C_6. (g) C_7.

classifier in the first concept and using it to detect the follow-up cyber-attacks, most of them are wrongly detected as reconnaissance when the concept drift occurs. For example, in Fig. 5(c), more than 98% DoS are identified as reconnaissance incorrectly. Once we discover a drift point, a classifier is retrained, Figs. 6 and 7 show the results. They verify that the detection performance improved greatly compared to the fixed training model. It reveals that incorporating concept drift analysis is promising and meaningful for cyber-attack detection on Bot-IoT.

V. Conclusion

Data streams were a core concept in IIoT. Sensor values were typically streamed to other devices and computational platforms. Corrupted data may be hard to detect and may reveal an underlying cyber-attack.

Data streams exposed several features that make their analysis a challenging task. Among those features, the most important were potential infinity (a stream should be analyzed under the assumption that it never ends) and evolution in time, known as drift or concept drift. More specifically, concept drift makes detection of attacks much harder. Both concept drift and attacks manifest themselves as changes in the statistical properties of the data stream, however, concept drifts were considered normal behavior, while the latter should produce an alert. In our IoT scenarios, an alert may indicate the presence of a cyber-attack. The presence of a concept drift makes many standard machine learning techniques unusable.

In this article, the authors provided an approach abbreviated as DRPM, based on statistical methods for concept drift analysis, which is suitable for attack detection in data streams. The evaluation of proposed techniques on synthetic and realistic dataset proves the ability to detect concept drift. The DRPM model was capable of detecting the drift points more accurately on the datasets of SEA and UG-2C-5D with lower $Delay$. We design a subdataset that includes five different concepts to represent realistic cyber-attack situations in IoT scenarios. With the help of concept drift analysis, the performance of cyber-attack detection obtained by CNN and LSTM on the Bot-IoT were improved significantly in the various metrics. We should point out that the performances affected by FN and larger $Delay$ of drift points were more severe than FP on Bot-IoT.

This was successfully used for cyber-attack discovery in IoT. Specifically, it was useful since data labels were often invalid in data streams and the tag information was unnecessary in DRPM. The inherent feature of residual projection results in limitations, i.e., the number of principal components was relatively sensitive to the performance. Another direction for improvement was to effectively determine the capacity of the dynamic window, as well as the fixed window.

For future work, the authors plan to perform more extensive evaluation, including multidimensional feature spaces, and refine the model and techniques using more sophisticated statistical methods and tools. We are also considering more applications of concept drift analysis such as predictive maintenance and quality inspection of materials.

References

[1] E. Benkhelifa, T. Welsh, and W. Hamouda, "A critical review of practices and challenges in intrusion detection systems for IoT: Toward universal and resilient systems," *IEEE Commun. Surv. Tut.*, vol. 20, no. 4, pp. 3496–3509, Oct.–Dec. 2018.

[2] J. Arshad, M. A. Azad, R. Amad, K. Salah, M. Alazab, and R. Iqbal, "A review of performance, energy and privacy of intrusion detection systems for IoT," *Electronics*, vol. 9, pp. 1–24, 2020.

[3] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and Kuang-YuanTung, "Intrusion detection system: A comprehensive review," *J. Netw. Comput. Appl.*, vol. 36, no. 1, pp. 16–24, 2013.

[4] J. Huang, L. Kong, G. Chen, M.-Y. Wu, X. Liu, and P. Zeng, "Towards secure industrial IoT: Blockchain system with credit-based consensus mechanism," *IEEE Trans. Ind. Informat.*, vol. 15, no. 6, pp. 3680–3689, Jun. 2019.

[5] H. Qiao, J. O. Blech, and H. Chen, "A machine learning based intrusion detection approach for industrial networks," in *Proc. IEEE Int. Conf. Ind. Technol.*, 2020, pp. 265–270.

[6] N. K. Kandasamy, "An investigation on feasibility and security for cyber-attacks on generator synchronization process," *IEEE Trans. Ind. Informat.*, vol. 16, no. 9, pp. 5825–5834, Sep. 2020.

[7] P. Guo, J. Cao, and X. Liu, "Lossless in-network processing in WSNs for domain-specific monitoring applications," *IEEE Trans. Ind. Informat.*, vol. 13, no. 5, pp. 2130–2139, Oct. 2017.

[8] I. Jawhar, N. Mohamed, J. Al-Jaroodi, and S. Zhang, "An architecture for using autonomous underwater vehicles in wireless sensor networks for underwater pipeline monitoring," *IEEE Trans. Ind. Informat.*, vol. 15, no. 3, pp. 1329–1340, Mar. 2019.

[9] M. D. Benedetti, F. Leonardi, F. Messina, C. Santoro, and A. Vasilakos, "Anomaly detection and predictive maintenance for photovoltaic systems," *Neurocomputing*, vol. 310, pp. 59–68, 2018.

[10] W. Yu, T. Dillon, F. Mostafa, W. Rahayu, and Y. Liu, "A global manufacturing Big Data ecosystem for fault detection in predictive maintenance," *IEEE Trans. Ind. Informat.*, vol. 16, no. 1, pp. 183–192, Jan. 2020.

[11] I. Žliobaitė, M. Pechenizkiy, and J. Gama, "An overview of concept drift applications," in *Big Data Analysis: New Algorithms for a New Society, Series Studies in Big Data*, N. Japkowicz and J. Stefanowski, Eds. New York, NY, USA: Springer, 2015, pp. 91–114.

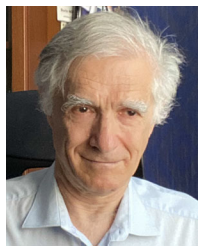
[12] S. Ramírez-Gallego, B. Krawczyk, S. García, M. Woźniak, and F. Herrera, "A survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017.

[13] N. Koroniotisa, N. Moustafaa, E. Sitnikovaa, and B. Turnbull, "Towards the development of realistic botnet dataset in the Internet of Things for network forensic analytics: Bot-IoT dataset," *Future Gener. Comput. Syst.*, vol. 100, pp. 779–796, 2019.

[14] F. Zhang, H. A. D. E. Kodituwakku, J. W. Hines, and J. Coble, "Multilayer data-driven cyber-attack detection system for industrial control systems based on network, system, and process data," *IEEE Trans. Ind. Informat.*, vol. 15, no. 7, pp. 4362–4369, Jul. 2019.
[15] F. Farivar, M. S. Haghighi, A. Jolfaei, and M. Alazab, "Artificial intelligence for detection, estimation, and compensation of malicious attacks in nonlinear cyber-physical systems and industrial IoT," *IEEE Trans. Ind. Informat.*, vol. 16, no. 4, pp. 2716–2725, Apr. 2020.
[16] O. A. Beg, T. T. Johnson, and A. Davoudi, "Detection of false-data injection attacks in cyber-physical DC microgrids," *IEEE Trans. Ind. Informat.*, vol. 13, no. 5, pp. 2693–2703, Oct. 2017.
[17] M. A. Khan and K. Salah, "Iot security: Review, blockchain solutions, and open challenges," *Future Gener. Comput. Syst.*, vol. 82, pp. 395–411, 2018.
[18] M. A. Ferrag, L. Maglaras, S. Moschoyiannis, and H. Janicke, "Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study," *J. Inf. Secur. Appl.*, vol. 50, 2020, Art. no. 102419.
[19] R. Vinayakumar, M. Alazab, S. Srinivasan, Q.-V. Pham, S. K. Padannayil, and K. Simran, "A visualized botnet detection system based deep learning for the Internet of Things networks of smart cities," *IEEE Trans. Ind. Appl.*, vol. 56, no. 4, pp. 4436–4456, Jul./Aug. 2020.
[20] M. Shafiq, Z. Tian, Y. Sun, X. Du, and M. Guizani, "Selection of effective machine learning algorithm and Bot-IoT attacks traffic identification for Internet of Things in smart city," *Future Gener. Comput. Syst.*, vol. 107, pp. 433–442, 2020.
[21] M. Shafiq, Z. Tian, A. K. Bashir, X. Du, and M. Guizani, "CorrAUC: A malicious Bot-IoT traffic detection method in IoT network using machine learning techniques," *IEEE Internet Things J.*, vol. 8, no. 5, pp. 3242–3254, Mar. 2021.
[22] N. Koroniotis, N. Moustafa, and E. Sitnikova, "A new network forensic framework based on deep learning for Internet of Things networks: A particle deep framework," *Future Gener. Comput. Syst.*, vol. 110, pp. 91–106, 2020.
[23] M. A. Ferrag and L. Maglaras, "DeepCoin: A novel deep learning and blockchain-based energy exchange framework for smart grids," *IEEE Trans. Eng. Manage.*, vol. 67, no. 4, pp. 1285–1297, Nov. 2020.
[24] S. I. Popoola, B. Adebisi, M. Hammoudeh, G. Gui, and H. Gacanin, "Hybrid deep learning for botnet attack detection in the Internet-of-Things networks," *IEEE Internet Things J.*, vol. 8, no. 6, pp. 4944–4956, Mar. 2021.
[25] Y. Jia, F. Zhong, A. Alrawais, B. Gong, and X. Cheng, "FlowGuard: An intelligent edge defense mechanism against IoT ddos attacks," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 9552–9562, Oct. 2020.
[26] L. Bu, C. Alippi, and D. Zhao, "A pdf-free change detection test based on density difference estimation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 99, no. 2, pp. 324–334, Feb. 2018.
[27] A. Yazidia, J. Oommen, G. Hornc, and O.-C. Granmo, "Stochastic discretized learning-based weak estimation: A novel estimation method for non-stationary environments," *Pattern Recognit.*, vol. 60, pp. 430–443, 2016.
[28] G. Ditzler, M. Roveri, C. Alippi, and R. Polikar, "Learning in nonstationary enviroments: A survey," *IEEE Comput. Intell. Mag.*, vol. 10, no. 4, pp. 12–25, Nov. 2015.
[29] D.-S. Pham, S. Venkatesh, M. Lazarescu, and S. Budhaditya, "Anomaly detection in large-scale data stream networks," *Data Mining Knowl. Discov.*, vol. 28, pp. 145–189, 2014.
[30] V. M. A. Souza, D. F. Silva, J. Gama, and G. E. A. P. A. Batista, "Data stream classification guided by clustering on nonstationary environments and extreme verification latency," in *Proc. SIAM Int. Conf. Data Mining*, 2015, pp. 873–881.
[31] H. Yang and S. Fong, "Moderated VFDT in stream mining using adaptive tie threshold and incremental pruning," in *Proc. Int. Conf. Data Warehousing Knowl. Discov.*, 2011, pp. 471–483.
[32] R. Kirkby, "Improving hoeffding trees," Ph.D. dissertation, Dept. Comput. Sci., Univ. Waikato, Hamilton, New Zealand, 2007.
[33] I. Ali *et al.*, "Systematic literature review on IoT-based botnet attack," *IEEE Access*, vol. 8, pp. 212 220–212232, 2020.
[34] E. Bertino and N. Islam, "Botnets and Internet of Things security," *Computer*, vol. 50, no. 2, pp. 76–79, 2017.
[35] S. G. Abbas, S. Zahid, F. Hussain, G. A. Shah, and M. Husnain., "A threat modelling approach to analyze and mitigate botnet attacks in smart home use case," in *Proc. 14th Int. Conf. Big Data Sci. Eng.*, 2020, pp. 122–129.
[36] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE Commun. Surv. Tut.*, vol. 21, no. 3, pp. 2671–2701, Jul.–Sep. 2019.
[37] S. Mahdavifar and A. A. Ghorbani, "Application of deep learning to cybersecurity: A survey," *Neurocomputing*, vol. 347, pp. 149–176, 2019.
[38] O. I. Abiodun, A. Jantan, A. E. Omolara, K. V. Dada, N. A. Mohamed, and H. Arshad, "State-of-the-art in artificial neural network applications: A survey," *Heliyon*, vol. 4, no. 11, 2018, Art. no. e00938.
[39] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.

Hanli Qiao received the Ph.D. degree in mathematics from the University of Turin, Turin, Italy, in 2017.

She is currently working with the College of Science, Guilin University of Technology, Guilin, China. She has been working in various realms, and her research interests include IoT security under nonstationary environments, concept drift detection, machine- and deep learning applications, and pattern recognition.



Boris Novikov (Member, IEEE) received the graduate degree in mathematics from the School of Mathematics and Mechanics, Leningrad University, Saint Petersburg, Russia, in 1972.

Since 1991, he has been with Operations Research Laboratory, Saint-Petersburg University. In 2000, he was a Professor, and from 2013 to 2018, the Department Chair with Saint Petersburg University. In 2019, he joined National Research University "Higher School of Economics." He visited CRAI, Italy, and Aalto University, Espoo, Finland, was a leading Researcher for several projects funded by INTS and Russian agencies, as well as industrial partners. He is currently a Professor with the Department of Informatics, National Research University Higher School of Economics, Saint Petersburg, Russia. His research interests include information management and several aspects of design, development and tuning databases, applications, and database management systems, as well as distributed scalable systems for stream processing and analytics.

Prof. Novikov is a Member of journal editorial boards of *Programming and Computer Software" and "Computer Science and Information Systems*, and was a Program Committee Chair for several international conferences. He was a Member of IEEE CS and ACM.

Jan Olaf Blech (Member, IEEE) received the Ph.D. degree in computer science from the University of Kaiserslautern, Kaiserslautern, Germany, in 2009.

He was a Professor of practice with Aalto University, Espoo, Finland. Before joining Aalto as a Professor of practice with the Factory of the Future, he worked as an expert in industry. Prior to this, he worked in different research centres and institutes in Australia (2013–2017), Germany (fortiss GmbH, Munich: 2010–2013) and France (Verimag: Grenoble 2008–2010). His research interests included factory automation, embedded systems, software architecture, and formal methods. Sadly, he passed away in February 2021.