Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.035126
Article

Concept Drift Analysis and Malware Attack Detection System Using Secure Adaptive Windowing

Emad Alsuwat[1,*], Suhare Solaiman[1] and Hatim Alsuwat[2]

[1]Department of Computer Science, College of Computers and Information Technology, Taif University, Taif, 26571, Saudi Arabia
[2]Department of Computer Science, College of Computer and Information Systems, Umm Al-Qura University, Makkah, 24382, Saudi Arabia
*Corresponding Author: Emad Alsuwat. Email: Alsuwat@tu.edu.sa
Received: 08 August 2022; Accepted: 29 January 2023

Abstract: Concept drift is a main security issue that has to be resolved since it presents a significant barrier to the deployment of machine learning (ML) models. Due to attackers' (and/or benign equivalents') dynamic behavior changes, testing data distribution frequently diverges from original training data over time, resulting in substantial model failures. Due to their dispersed and dynamic nature, distributed denial-of-service attacks pose a danger to cybersecurity, resulting in attacks with serious consequences for users and businesses. This paper proposes a novel design for concept drift analysis and detection of malware attacks like Distributed Denial of Service (DDOS) in the network. The goal of this architecture combination is to accurately represent data and create an effective cyber security prediction agent. The intrusion detection system and concept drift of the network has been analyzed using secure adaptive windowing with website data authentication protocol (SAW_WDA). The network has been analyzed by authentication protocol to avoid malware attacks. The data of network users will be collected and classified using multilayer perceptron gradient decision tree (MLPGDT) classifiers. Based on the classification output, the decision for the detection of attackers and authorized users will be identified. The experimental results show output based on intrusion detection and concept drift analysis systems in terms of throughput, end-end delay, network security, network concept drift, and results based on classification with regard to accuracy, memory, and precision and F-1 score.

Keywords: Concept drift; machine learning; DDOS; cyber security; SAW_WDA; MLPGDT

1 Introduction

The current technological world of present era is changing and making it harder to protect systems and links against mischievous attacks or breaches. One sort of security technology is called an intrusion

detection system (IDS) that has been designed to identify and prevent intrusions in a network system. Because it contains such a big amount of data and information, the Internet has a variety of issues in terms of making it a secure system. Businesses, industries, and different spheres of daily activity all use computer networks. Organizations and institutions all over the world have been obliged to build and employ modern networks for safety as a result of technological and business advancements [1].

A shift in the features of the data stream is known as concept drift. When properties of decision attributes as well as classes to be forecasted vary unexpectedly between two given time points, it is defined as concept drift. Classification quality may suffer as a result of this circumstance and learning mechanisms may suffer as a result [2]. In ML, concept drift relates to a shift in relationships between input and output data in a data stream. Data could be altered in any way. Other sorts of changes include (i) gradual changes over time, (ii) recurring or cyclical changes, and (iii) abrupt or sudden changes. Learning models must be able to adjust to changes swiftly and accurately. The ideal drift detection approach is used to detect incoming new communications autonomously. The drift detector appears to be the simplest classifier, however, it is not as straightforward as it appears. The model should usually be rebuilt as soon as feasible after returning the signal regarding the drift [3].

Learning techniques in embedded applications have been required by recent improvements in cyber-physical systems (CPS) to work in non-stationary, time-variant contexts [4]. Idea drift learning, sometimes referred to as learning in non-stationary contexts, concentrates on the environment's event-driven changes in CPS. Changes in feature data (x) and goal variables (y) altered underlying models developed by learning methods as a result of such evolving notions. Concept drift detection in CPS reduces negative compounding error impact and allows for cost-effective predictive maintenance [5]. In this setting, ensemble learning algorithms that include many supervised techniques determine it impossible as well as impractical to detect concept drifts.

A novel unsupervised ML method is required to solve these issues and to manage complicated data patterns as well as distributional assumption breaches buried in industrial applications of CPS data streams. In supervised ML problems, a machine learning classifier is trained using a given labeled dataset of training samples with the goal of predicting a target variable. Concept drift in this situation refers to the alteration over time of the relationship between the input data and the target variables.

Concept drift may emerge in dynamic environments, such as e-mail spam detection. In this dynamic environment, malicious opponents may attempt change their e-mails to avoid spam filters. Ineffective classifiers are unable to accurately categorize newer samples as a result of these changes in data distribution. necessitating the development of algorithms for responding to concept drifts [6].

The following are the chief contributions of this study paper:

- To design novel architecture in concept drift analysis and detection of malware attacks like DDOS in the network
- The network has been analyzed for detecting the intrusion and concept drift using secure adaptive windowing with website data authentication protocol (SAW_WDA) integrated with authentication protocol to avoid DDOS attacks and concept drift.
- The user data of the network will be collected and classified using multilayer perceptron gradient decision tree (MLPGDT) classifiers.
- Based on the classification output, the decision for the detection of attackers and authorized users will be identified.

The model of this essay is organized as follows. Section 2 of our report includes the associated work. Section 3 of our proposal presents the system model. Performance analysis is presented in Section 4. In Section 5, the conclusion of our research is presented.

2 Related Work

Idea drift is pertinent for malware detection when static file analysis is performed, according to earlier research [7]. Prior studies have looked into methods for identifying idea drift in malware families [8] and warning human analysts when it is found during malware detection. The efficiency of several machine learning properties for detecting fraudulent websites is examined in work [9] However, the use of Host and Content capabilities is the extent of their activity. Extract the Lexical features, as well as the Host and With features based on content, from each URL and then keep them in feature vector form. The supervised learning system uses these feature vectors as input to classify these URLs as harmful or benign. Random Forest (RF), Gradient Boosted Trees (GBT), and Feed Forward Neural Networks (FFNN) have supervised learning methods employed in our research. They use unsafe databases and benign URLs collected from diverse sources to train their algorithms. Although the suggested method is flexible and resistant to a range of dangers, it disregards the dynamic nature of websites. The same training data reaches the hands of criminals and the ability to spot patterns in detection methods tries to change some elements of harmful websites to go around the security [10].

Among the machine learning techniques suggested in [11] for identifying fake websites are (Lagrangian Support Vector Machine) LSVM, (Logistic regression) LR, Random Forest (RF), Naive Bayes (NB), and statistical techniques for finding Concept Drifts in websites. Only a few studies have produced practical results, according to the author in [12], intending to foster research on intelligent security techniques based on a cyclic process that begins with the discovery of new threats and ends with the analysis and development of prevention measures. The authors of [13] propose a novel Gradient Boosting Decision Tree (GBDT) training technique with narrower sensitivity limitations and much better noise allocations. To slight the sensitivity boundaries by analyzing the gradient characteristic and the contributions of each tree in GBDTs, they suggest flexibly regulate the gradients of training data for every iteration and leaf node clipping Furthermore, they develop a unique boosting structure to distribute the privacy budget among trees, reducing the precision loss even further. Their studies reveal that our technique outperforms other baselines in terms of model accuracy. Jiang et al. conducted a comprehensive review of several articles that used ML in security domains, resulting in a taxonomy of machine learning models and their applications in cybersecurity [14]. Because label data is rarely available in real-world applications, [15] classified existing solutions for detecting abnormalities in changing data using unsupervised algorithms. The research [16] looked at adversarial assaults on PDF malware detectors.

The state-of-the-art of ML for data streams was emphasized by the author in [17], who presented possible research options. A lot of research is not valid in many use situations, according to [18], which focuses on label acquisition and model deployment. In [19], the author conducted a comparative examination of several methodologies for dealing with imbalanced data, applying them to various data distributions and application domains. [20] Investigated some of the limitations and challenges of Deep Learning (DL) methods in a traditional ML workflow for malware detection as well as classification in literature such as open benchmarks, class imbalance, concept drift, model interpretability, and adversarial learning.

3 System Model

The novel design in concept drift analysis and malware attack detection is covered in this section. Here the website with concept drift has been predicted and analyzed for DDOS attack in the network. When the data drift is detected, the web server has been analyzed and predicted to be concept drift. Then SAW_WDA has been employed for minimizing the concept drift and intrusion of the network. The data of users has been collected and classified using MLPGDT where the decision has been made based on classified output whether the malware user or authorized user. Fig. 1 depicts the total system architecture.

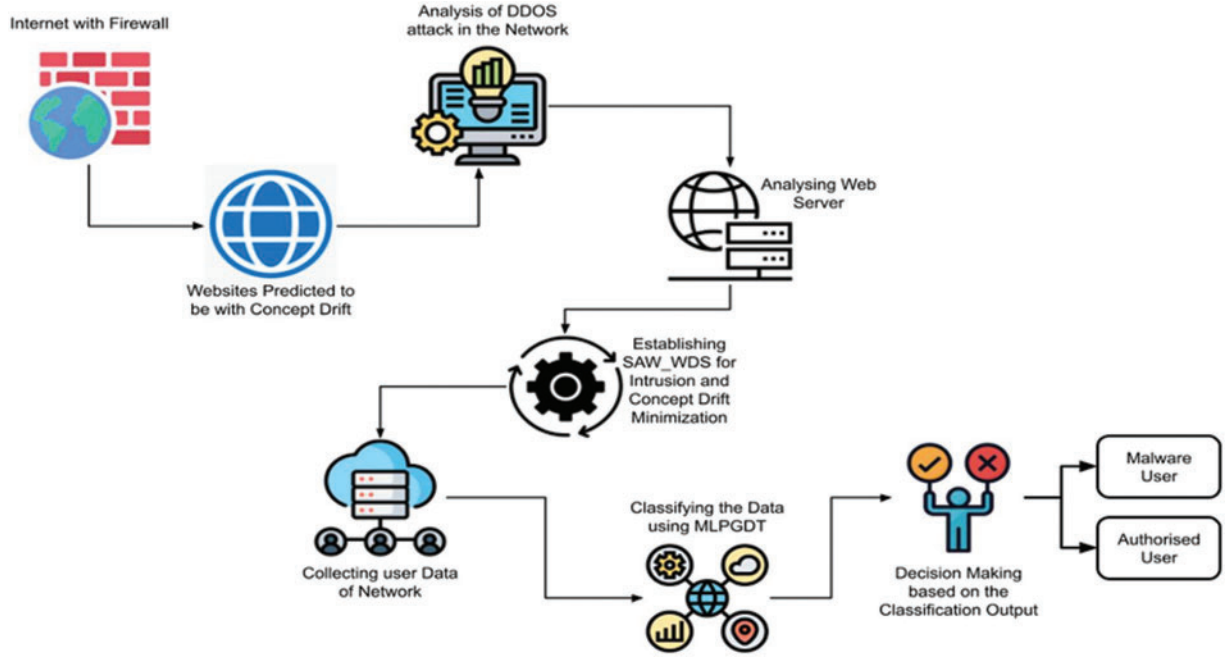


Figure 1: shows the proposed systems' overall architecture

3.1 Secure Adaptive Windowing with Website Data Authentication Protocol (SAW_WDA)

The intrusion detection system and concept drift of the network has been analyzed using secure adaptive windowing with website data authentication protocol (SAW_WDA). To detect the adaptive windowing change, we have performed the SAW_WDA protocol. The detailed SAW_WDA protocol is given below:

When no obvious change is found, the window is dynamically magnified, and when a change is determined it is compressed. In section W0 · W1 of W, the cut value is evaluated as under. Let W stand for W length, $\hat{}\mu$W for the average of W's elements, and W for the average of μt for t ∈ W. Let n0 show the size of W0, n1 show the size of W0, and W1 and n show the length of W, resulting in n = n0 + n1.

W0 and the predicted values are characterized by W1 To achieve the most stringent performance guarantees are given by Eq. (1)

$$\begin{aligned} m &= \frac{1}{1/n_0 + 1/n_1} \text{ (harmonic mean of } n_0 \text{ and } n_1) \\ \delta' &= \frac{\delta}{n}, \text{ and } \epsilon_{cut} = \sqrt{\frac{1}{2m} \cdot \ln \frac{4}{\delta'}} \end{aligned} \tag{1}$$

If we undertake S as a data stream and E is an ensemble. When the case is received, the internal change detector D is used to train the online classifier gradually. The ensemble member's $Ci \in E$ are weighted after every incoming instance, rather than calculating component classifiers by Eq. (2).

$$W_o = \sum\nolimits_{1 \le t \le T} W\left(s_c(P(t))\right), W_e = W\left(s_c(C)\right) - W_o \tag{2}$$

$$W\left(s_c\left(P\left(t_1\right)\right) \cap s_c\left(P\left(t_2\right)\right)\right) = a_1$$

For all $1 \le t_1, t_2 \le T$

$$A_1 = \begin{cases} W\left(s_c\left(P\left(t_1\right)\right)\right) & \text{if } t_1 = t_2 \\ 0 \text{ if } t_1 \ne t_2 \end{cases}$$

$$A\left(s_c\left(P_{ls}\left(t_1\right)\right) \cap s_c\left(C_f\right)\right) + \sum\nolimits_{1 \le t_2 \le T, t_2 \ne t_1} A\left(s_c\left(P_{ls}\left(t_1\right)\right) \cap s_c\left(P_{us}\left(t_2\right)\right)\right) \ge 0.6 \times A\left(s_c\left(P_{ls}\left(t_1\right)\right)\right)$$

For all $1 \le t_1 \le T$

$$W\left(s_c\left(P\left(t_1\right)\right) \cap s_c\left(C\right)\right) = W\left(s_c\left(P\left(t_1\right)\right)\right) \tag{3}$$

For all $1 \le t_1 \le T$

When the underlying function that creates instances evolves, concept drift is said to occur. Formally, it can be described as any situation where the joint probability shifts. Thus, it may appear as a change in the class prior probabilities, a change in the class-conditional PDF, or a combination of the two. The transition probability of road segment Rj at time $t-1$ is used to compute the under-mention method that can enter road segment Ri at time t given by Eq. (4). Here 'traffic' denotes the average network traffic.

$$\Pr\left(R_j \text{ at } t-1 \mid R_i \text{ at } t\right) = p_j = \frac{\text{traffic}_j}{\sum_{p=1}^{n} \text{traffic}_p}$$

$$S_c(P(t)) = \left\{(\overleftarrow{x}, \overleftarrow{y}, \overleftarrow{z}) \in R^3 \mid p_{x,p(t)} \le \overleftarrow{x} \le p_{x,P(t)} + l_{x,P(t)}, p_{y,P(t)} \le \overleftarrow{y} \le p_{y,P(t)} + l_{y,P(t)}, p_{z,P(t)} \le \overleftarrow{z} \le p_{z,P(t)} + l_{z,P(t)}\right\}$$

$$s_c© = \left\{(\overleftarrow{x}, \overleftarrow{y}, \overleftarrow{z}) \in R^3 \mid 0 \le \overleftarrow{x} \le L_x, 0 \le \overleftarrow{y} \le L_y, 0 \le \overleftarrow{z} \le L_z\right\}$$

$$s_c\left(C_f\right) = \left\{(\overleftarrow{x}, \overleftarrow{y}, \overleftarrow{z}) \in R^3 \mid 0 \le \overleftarrow{x} \le L_x, 0 \le \overleftarrow{y} \le L_y, \overleftarrow{z} = 0\right\}$$

$$s_c\left(C_f^h\right) = \left\{(\overleftarrow{x}, \overleftarrow{y}) \in R^2 \mid (\overleftarrow{x}, \overleftarrow{y}) \in s_c\left(C_f\right)\right\} \tag{4}$$

Assume that E is an ensemble and S is a data stream. When the case is received, the internal change detector D is used to train the online classifier gradually. Each incoming instance results in a weighting of the ensemble members Ci and E, rather than calculating component classifiers by Eq. (2).

$$E = -\sum_{j} \left(p_j \cdot \log p_j\right) \tag{5}$$

The Hoeffding bound asserts that the estimated mean will not deviate from the true mean by more than with probability 1 by Eq. (6) after n independent observations of range R.

$$E = \frac{R^2 \ln(1/\delta)}{2n} \tag{6}$$

where a user-defined confidence parameter $\delta \in (0, 1)$ is used. Let's call the two subwindows W_0 and W_1. With $1-\delta$ probability, we obtains $|\mu W_0 - \mu W_1| \leq 2\varepsilon$.

Such that ε is Hoeffding bound while μW_0 and μW_1 are average of two sub-windows.

$$P(t): p_{P(t)} = \left(p_{I,P(t)}, p_{j,P(t)}, p_{k,P(t)}\right) \in I^3_{++}, l_{P(t)}$$

$$= \left(l_{I,P(t)}, l_{j,P(t)}, l_{k,P(t)}\right) \in I^3_{++}$$

$$L_{I,P(t)} = \frac{l_{x,P(t)}}{l_x}, l_{j,P(t)} = \frac{l_{y,P(t)}}{l_y}, l_{k,P(t)} = \frac{l_{z,P(t)}}{l_z}$$

$$L_i = \frac{L_x}{l_x}, L_j = \frac{L_y}{l_y}, L_k = \frac{L_z}{l_z} \tag{7}$$

$\mathbb{I}^n_{++}$ is *the n*-dimensional space of positive integers.

Assume that W's true mean is μ. $|\mu W_0 - \mu| \leq \varepsilon$ and $|\mu W_1 - \mu| \leq \varepsilon$ might be obtained independently, according to the Hoeffding bound. After that, they can be turned into $-\varepsilon \leq |\mu W_0 - \mu| \leq \varepsilon$ and $-\varepsilon \leq |\mu - \mu W_1| \leq \varepsilon$. In addition, these two inequalities together by Eq. (8).

$$\left|M_{W_0} - \mu_{W_1}\right| \leq 2\varepsilon$$

$$\left|\mu_{W_0} - \mu_{W_1}\right| \leq \sqrt{\frac{2R^2 \ln\left(\frac{1}{\delta}\right)}{n}} \tag{8}$$

Two equal-length sub windows made up sliding window W. From W_L to W_R, the Kullback-Leibler distance is given by Eq. (9).

$$KL\left(W_L \parallel W_R\right) = \sum_{x \in X} p_{W_L}(x) \log \frac{p_{W_L}(x)}{p_{W_R}(x)} \tag{9}$$

We make sure that the sum is calculated over X atoms (in a discrete setting, X is the event space). When the distance exceeds the threshold calculated using, a change is recognized (4). Then window WL's older portion is taken out.

s is the discrete-window analog of the operator sc given by Eq. (10)

The method takes as inputs a confidence value (0, 1) and a sequence of real values x1, ... , xt. Note that, the value of xt is only accessible at time point t. According to a certain Dt distribution, each xt

is independently formed for each t. Indicate the anticipated value and variance with t and t2t when xt is drawn following Dt.

$$S\left(P_{Ls}\left(t\right)\right)=\left\{\left(\overleftarrow{t},\overleftarrow{J},\overleftarrow{k}\right)\in I_{++}^{3}\mid p_{I,P(t)}\leq\tau\leq p_{I,p(t)}+l_{I,p(t)}-1,p_{j,p(t)}\leq\overleftarrow{j}\right. \tag{10}$$

$$\left.\leq p_{j,p(t)}+l_{j,P(t)}-1,\overleftarrow{k}=p_{k,p(t)}\right\}$$

$$S\left(P_{ls}^{h}(t)\right)=\left\{(\overleftarrow{i},\overleftarrow{j})\in I_{++}^{2}\mid\left(\overleftarrow{I},\overleftarrow{J},p_{k,P(t)}\right)\in s\left(P_{Is}(t)\right)\right\}$$

$$S\left(P_{us}\left(t\right)\right)=\left\{\left(\overleftarrow{1},\overleftarrow{j},\overleftarrow{k}\right)\in I_{++}^{3}\mid p_{I,P(t)}\leq\overleftarrow{1}\leq p_{I,p(t)}+l_{I,p(t)}-1,p_{j,p(t)}\leq\overleftarrow{j}\right.$$

$$\left.\leq p_{j,p(t)}+l_{j,p(t)}-1,\overleftarrow{k}=p_{k,p(t)}+l_{k,p(t)}-1\right\}$$

$$S\left(C\right)=\left\{\left(\overleftarrow{I},\overleftarrow{j},\overleftarrow{k}\right)\in I_{++}^{3}\mid1\leq\overleftarrow{I}\leq L_{I},1\leq\overleftarrow{j}\leq L_{j},1\leq\overleftarrow{k}\leq L_{k}\right\}$$

$$S\left(C_{f}^{h}\right)=\left\{(\overleftarrow{1},\overleftarrow{j})\in I_{++}^{2}\mid(\overleftarrow{1},\overleftarrow{j},1)\in s\left(C_{f}\right)\right\} \tag{11}$$

One can infer that the window will begin to contract after "O" ("ln(1/)/" "" "2") steps if t has been fixed at a value for a long time and suddenly changes to a value.

$$\rho_{i,j}(t)=\begin{cases}\max\left\{\rho_{t,}(t-1)\mid(\tau,\overleftarrow{J})\in s\left(P_{ls}^{h}(t)\right)\right\} & \cdots\begin{cases}+l_{k,P(t)} & \text{for }(i,j)\in s\left(P_{ls}^{h}(t)\right)\\ & \text{for }(i,j)\notin s\left(P_{ls}^{h}(t)\right)\end{cases},\\ \rho_{i,j}(t-1)\end{cases}$$

$$\rho_{I,j}(0)=0\,\text{for}\,(I,j)\in s\left(C_{f}^{h}\right)$$

$$\rho_{I,j}(t)\leq L_{k}\,\text{for}\,(I,j)\in s\left(P_{Ls}^{h}(t)\right)$$

$$\text{Num}\left(\left\{(\overleftarrow{i},j)\in s\left(P_{ls}^{h}(t)\right)\mid\rho_{tJ}(t-1)\geq\rho_{I,j}(t-1)\ \text{ for }(I,j)\in s\left(P_{ts}^{h}(t)\right)\right\}\right)\geq0.6\times l_{I,P(t)}\times l_{j,P(t)}$$

$$\frac{e-\sum_{k=1}^{MaxW-1}a_{k}}{h}=\frac{h^{2}+(3-\ MaxW)h+1-\ MaxW\ +(-1/h)^{MaxW-1}}{h(h+1)^{2}}\epsilon$$

Proof: MaxW refers to the window size at time point $t=t_{MaxW}$. Thus, the $\left(e-\sum_{k=1}^{MaxW-1}a_{k}\right)/h$ is the privacy budget at $t=t_{MaxW}.\left(e-\sum_{k=1}^{MaxW-1}a_{k}\right)/h$ and $\sum_{k=1}^{MaxW-1}a_{k}=\sum_{k=1}^{Max-1}\{\epsilon/h(h+1).(-1/h)^{k-1}+\epsilon/(h+1)\}$. After evaluation,

$$\sum_{k=1}^{MaxW-1}a_{k}=\frac{\epsilon}{h\left(h+1\right)}\sum_{k=1}^{MaxW-1}\left\{\left(-\frac{1}{h}\right)^{k-1}+h\right\}$$

$$=\frac{\epsilon}{h(h+1)}\left\{\frac{1-(-1/h)^{MaxW-1}}{1+1/h}+h\cdot(MaxW-1)\right\} \tag{12}$$

Thus,

$$\frac{e-\sum_{k=1}^{Max-1} a_k}{h} = \frac{h^2 + (3 - MaxW)h + 1 - MaxW + (-1/h)^{MaxW-1}}{h(h+1)^2}\epsilon \tag{13}$$

Algorithm: SAW_WDA

Input:
If S: stands for the stream of data, C_0: stands for the online machine learning learner, D: change detector (adaptive windowing detector), d: is size of the working buffer; k: quantity of ensemble participants, B; long-term
Output:
E: is the ensemble of classifiers with weight k
For each example $x_1 \in S$ **do**
Use incremental training approach to train C_0 and D with x_1;
$B \leftarrow B \cup \{x_1\}$
if detecting change $==$ True $\| \ |B| = d$ **then**
Use B to build C';
Weight the new classifier C'
weight each classifier C_I (in ensemble);
if $|E| < k$ **then**
$E - E \cup \{C'\}$.
else
Let C' be the weakest ensemble member
$C_0 = B$ (reinitialization)
reinitialize D
$B \leftarrow \phi$
End For
Maximum Window Size MaxW
OUTPUT: Decaying Factor h
Set $f(h, MaxW) = \left(h^2 + (3 - MaxW)* h + 1 - MaxW + (-1/h)^{Max-1}\right)/h* (h+1)^2$
Set $L = 1$ and $R = MaxW - 1$
Set $h = MaxW - 1$
While $L < R$ **do**
Set $M = (L + R)/2$
If the value of the function f(mid, MaxW) > 0 **then**
if $h > M$ **then**
Set $h = M$ and $R = M$
else
Return h
End While

3.2 Multilayer Perceptron Gradient Decision Tree (MLPGDT)

Consider an $M - 1$ intermediate layer and one final output layer in a multi-layered feed-forward arrangement. The equivalent output for each layer for a given input data x is in Rdi, where $i \in \{0, 1, 2. . . M\}$. We aim to learn Fi: $Rdi-1 \rightarrow Rdi$ mappings for each layer i where the value if $i > 0$. The final output oM aims to minimize empirical loss L on the training dataset. The loss L is commonly

calculated using mean squared errors or cross-entropy with additional regularization components. Backpropagation can be used to efficiently complete such learning tasks when each Fi is parametric and differentiable. The chain rule is used to evaluate gradients of the loss function concerning each specification at each layer, and then gradient descent is used to update the parameters. The output for intermediate layers is considered the model's new representation once training is completed.

Formally, MLPGDT lowers the following regularized objective by Eq., given a convex loss function l and a training dataset. This given dataset has n cases (tuples) and d features.

$$\grave{L} = \sum_{i} l(\grave{y}_i, y_i) + \sum_{k} \Omega\,(f_k)$$

$$y_j(p) = f\left(\sum_{i=1}^{n} x_i(p) \cdot w_{ij} - \theta_j\right) \tag{14}$$

where a regularisation term is "(f) = " "1" /"2" "V" "" "2". The λ regularization parameter is R, and the leaf weight is V. At the t-th iteration, GBDT minimizes the following objective function by forming an approximation function of the loss by Eq. (15).

$$\grave{L}^{(t)} = \sum_{i=1}^{n} \left[g_i f_t\,(\mathbf{x}_i) + \frac{1}{2} f_t^2\,(\mathbf{x}_i)\right] + \Omega\,(f_t)$$

$$Y_k(p) = f\left(\sum_{i=1}^{m} x_{jk}(p) \cdot w_{jk}(p) – \theta_k\right)$$

$$E = E + \frac{(e_k(p))^2}{2} \tag{15}$$

$$G\,(I_L, I_R) = \frac{\left(\sum_{i \in I_L} g_i\right)^2}{|I_L| + \lambda} + \frac{\left(\sum_{i \in I_R} g_i\right)^2}{|I_R| + \lambda} \tag{16}$$

To discover the split that optimizes the gain, GBDT explores all feature values. If the current node fails to match the splitting criteria, With the best leaf value provided by Eq, it turns into a leaf node (17).

$$V(I) = -\frac{\sum_{i \in I} g_i}{|I| + \lambda} \tag{17}$$

Each A shrinkage rate is frequently applied to the leaf values, similar to the learning rate η in stochastic optimization, to reduce the effects of every specific tree and make room for subsequent trees to be enhanced model. Let f be a randomized function and ε be a positive real number. If given datasets, say D and D0, differ in at least one case or tuple and differ in any output O of function f, the function f is said to give -differential privacy as given by Eq. (18).

$$Pr[f(D) \in O] \leq e^{c} \cdot Pr\left[f\left(D'\right) \in O\right] \tag{18}$$

Here ε denotes the secrecy budget. The Laplace and exponential methods are commonly utilized to obtain ε-differential privacy by summing noise calibrated to a function's sensitivity. Error gradients for neurons in the output layer are calculated as follows (19):

$$\delta_k(p) = f \cdot e_k(p) \tag{19}$$

where f ' is the derived function for activation and error $e_k\,(p) = y_{d,k}\,(p) - y_k\,(p)$ is given by the following equations Eq. (20) through Eq. (22).

$$f'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = f(x)\cdot(1-f(x)) \tag{20}$$

$$f'(x) = \frac{2a\cdot e^{-a\cdot x}}{(1+e^{-a\cdot x})^2} = \frac{a}{2}\cdot(1-f(x))\cdot(1+f(x)) \tag{21}$$

$$\delta_k(p) = y_k(p)\cdot(1-y_k(p))\cdot e_k(p) \tag{22}$$

Weight gradients between the hidden and output layers have been updated by Eq. (23).

$$\Delta w_y(p) = \Delta w_j(p) + x_i(p)\cdot\delta_j(p)$$

$$w_{ij} = w_{ij} + \eta\cdot\Delta w_{ij} \tag{23}$$

This term is proportionate to the most recent weight modification, i.e., the values used to alter the weights are saved and have a direct impact on all subsequent adjustments by Eq. (24).

$$\Delta w_{ij}(p) = \Delta w_{ij}(p) + \alpha\cdot\Delta w_{ij}(p-1) \tag{24}$$

The variable learning rate technique [19] not only uses different learning frequency for each weight but also changes the learning parameters in each repetition based on the gradients' successive indications by Eq. (25).

$$\eta_p\,(p) = \begin{Bmatrix} u\cdot\eta_y(p-1), \operatorname{sgn}\left(wv_j(p)\right) = \operatorname{sgn}\left(bv_j(p-1)\right) \\ d\cdot\eta_y(p-1), \operatorname{sgn}\left(bv_\xi(p)\right) = -\operatorname{sgn}\left(w_j(p-1)\right) \end{Bmatrix} \tag{25}$$

We use the softmax extension of the single-pole sigmoid to classify the given n classes. Each class is corresponding to a binary output of the network as given by Eq. (26).

$$y_k = \frac{e^{y_k}}{\sum_{i=1}^{n} e^{y_i}} \tag{26}$$

Letting $g^* = \max_{I\in D}|g_I|$, we have $\Delta G \le 3g^{*2}\left(\sum_{i\in I_L} g_i\right)^2$. Let $\partial_{g_s} h = 0$ and $\partial_{\Sigma_{ee\sim I_L} g_i} h = 0$, we can get $g_s = 0$ and $\sum_{I\in I_L} g_I = 0$. Matching stationary point and border (i.e., $g_s = \pm g^*$ and $\sum_{I\in I_L} g_I = \pm n_i g^*$), determine $g_s = -g^*$, $\sum_{I\in I_L} g_I = n_i g^*$ (or $g_s = g^*$ and $\sum_{I\in I_2} g_I = -n_i g^*$) and $n_l \to \infty$, Eq. (9) can achieve maximum. Where I_f is the filtered instance set and I_f^c is remaining denote the approximation error of GDF on leaf values as $\xi_I = \left|V(I) - V\left(I_f^c\right)\right|$. Then, $\xi_I \le p\left(\left|\overleftarrow{g_f}\right| + g_l^*\right)$.

The incline of case xi is initialized as $g_I = \pm n_i g^*$ at the start of the training. Allow $g_l^* = \pm n_i g^*$ be the greatest feasible 1-norm gradient in the initialization. It's worth noting that $g * l$ is unaffected by training data and is solely dependent on the loss function l. The values of 1-norm gradients tend to drop as the number of trees in training rises because the loss function l is convex. As a result, the majority of examples have a smaller 1-average incline than g_l^* during the whole training procedure. As a result, use g_l^* threshold to filter training examples. Filter the cases that have a 1-norm gradient greater than g_l^* at the start of each iteration. In this cycle, just the lasting cases are used as input to create a new differentially private decision tree. It's worth noting that filtered illustrations may still be

used in the training of later trees. Assure that the gradients of the used instances are no bigger than g_l^* using a gradient-based data filtering strategy like this.

$$\xi_I = \left| \frac{\sum_{i\in I_f} g_i + \sum_{i\in I_f} g_i}{|I| + \lambda} - \frac{\sum_{i\in I_f} g_i}{(1-p)|I| + \lambda} \right|$$

$$= p\left| \frac{\sum_{i\in I_f} g_i}{|I_f| + p\lambda} \right| + p\left| \frac{|I|}{(|I| + \lambda)((1-p)|I| + \lambda)} \sum_{i\in I_j} g_i \right|$$

$$\leq \left| \frac{\sum_{i\in I_f} g_i}{|I| + \lambda} | + | \left(\frac{1}{|I| + \lambda} - \frac{1}{(1-p)|I| + \lambda} \right) \sum_{I\in I_f} g_i \right|$$

$$\leq p\left| \frac{\sum_{I\in I_f} g_i}{|I_f|} \right| + p\left| \frac{\sum_{I\in I_j} g_i}{(1-p)|I|} \right| \leq p\left(\left| \overleftarrow{g_f} \right| + g_i^* \right) \leq p\left(\left| \overleftarrow{g_f} \right| + g_i^* \right) \tag{27}$$

For the sake of simplicity, assume the instance's label is -1 and the gradient is g_l^*. Consider $V_1 = -\frac{g_i}{1+\lambda} \geq -g_i^*$ for the first tree. The enhancement in prediction value on the first tree is $-ng_l^*$, because the shrinkage rate is η by Eq. (28).

$$|V_2| \leq \left\| \frac{\partial l\,(-1, y)}{\partial y}\Big|_{y=\eta V_i} \right\| \approx \left\| \frac{\partial l(-1, y)}{\partial y}\Big|_{y=0} + \eta V_1 \right\| \leq g_i^*(1-\eta) \tag{28}$$

In the same way, $|V_1| \leq g_i^*(1-\eta)^{t-1}$ is obtained.

Algorithm: MLPGDT

Input:
M is the Number of Layers, di is layer dimension, X, and Y are training data, L is the final loss function, gamma, alpha, K1, K2, noise injection σ^2, epoch E
Output:
A trained mGBDT

$F_{1:M}^0 \leftarrow$ Assign $() ; G_{2:M}^0 \leftarrow$ Assign $() ; o_o \leftarrow X; o_j \leftarrow F_j^0\,(oj-1)$
where $j = 1, \ldots, M$
For $t = 1$ to E **do**
For $j = M$ to 2 **do**
$\mathbf{L_j^{inv}} \leftarrow \| \mathbf{G_j^t}\left(\mathbf{F_j^{t-1}}\left(\mathbf{o_{j-1}^{noine}}\right)\right) - \mathbf{o_{j-1}^{noine}} \|$

$$\mathbf{r_k} \leftarrow -\left[\frac{\partial \mathbf{L_j^{inv}}}{\partial \mathbf{G_j^t}\left(\mathbf{F_j^{t-1}}\left(\mathbf{o_{j-1}^{nase}}\right)\right)} \right]$$

Fit regression tree h_k to r_k, i.e. utilizing training set $\left(F_j^{t-1}\left(o_{j-1}^{noisc}\right), r_k\right)$
$G_j^t \leftarrow G_j^t + \gamma h_k$
$G_j^t\left(F_j^{t-1}\left(o_{j-1}^{noine}\right)\right)$
End For
$Z_{j-1}^t \leftarrow G_j^t\left(Z_j^t\right)$// evaluate pseudo-label for layer $j-1$

(Continued)

Algorithm: Continued

End For
For j=1 to M **do**
$F_j^t \leftarrow F_j^t$ // update F_j^t using Z_j^t. This update is for k2 rounds
For (k = 1; k <= k2; k++) **do**
$L_j \leftarrow \| F_j^t(o_{j-1}) - z_j^t \|$
$r_k \leftarrow -\left[\frac{\partial L_j}{\partial F_j(o_{j-1})}\right]$
Fit h_k to r_k, where $h_k$is the regression tree. We do this via dataset (o_{j-1}, r_k)
$F_j^t \leftarrow F_j^t + \gamma h_k$
End For
End For
End For
Return $F_{1:M}^0, G_{2:M}^0$

4 Performance Analysis

Python was used to test the suggested technique on Windows 10. In addition, the OpenCV library was used to recognize and test datasets. The following is a description of the dataset.

DDoS 2016: The data were obtained in a controlled setting and included four types of malicious network attacks: Hypertext Transfer Protocol (HTTP) Flood, SQL Injection Dos (SIDDOS), User Datagram Protocol (UDP) Flood, and Smurf. There are 27 characteristics, 5 classes, and 734,627 records in the dataset.

UNSW-NB15: It was created in a small network environment over a short period of time (31 h), using the IXIA Perfect Storm tool, and it combines real average network traffic activities with fictitious attack behaviors, producing 175,341 records for training and 82,332 records for testing.

IXIA tool was used to simulate nine different sorts of attacks. Basic, content, time, and further produced features based on statistical characteristics of connections are among the 49 features available for study in the dataset.

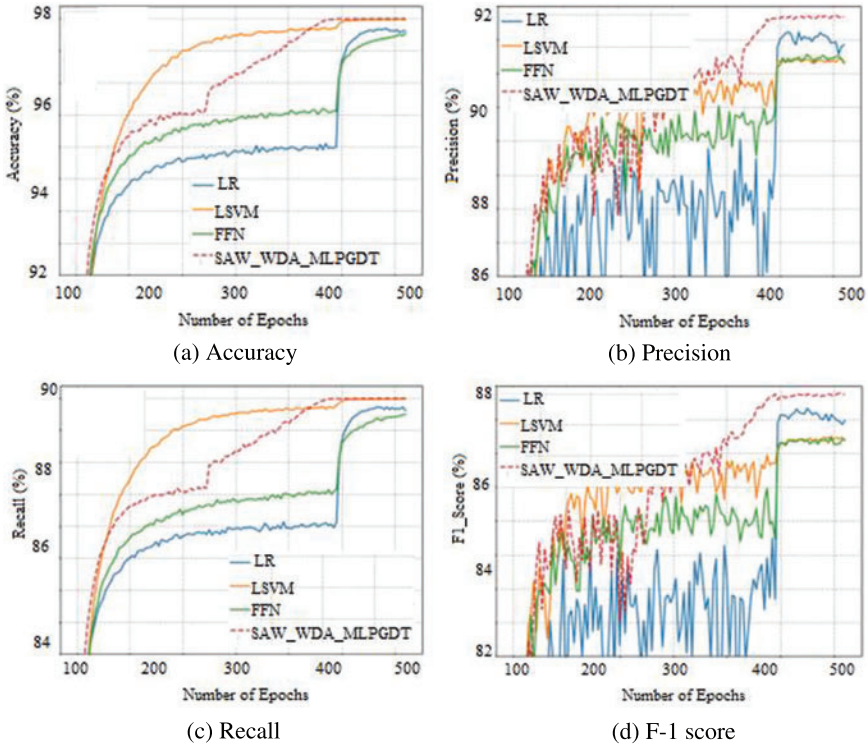
CICIDS 2017: The Canadian Institute for Cybersecurity (CIC) made the data available to the public. In the creation process, There are two different types of user profiles, multistage attacks like Heart bleed, and several DoS and DDoS attacks were used. The CICFlowMeter utility extracts 80 network traffic features from the data. The background traffic was generated using user profiles based on the abstract human behavior of 25 users using HTTP, File Transfer Protocol (FTP), HTTPS, Secure Shell (SSH), and email protocols. Traffic was created for a brief time (5 days).

The suggested strategy and existing methods are compared in Table 1 in terms of accuracy, precision, recall, and F-1 score. Here comparative analysis has been carried out for various datasets namely LR, LSVM, feed foreword neural network (FFNN), and SAW_WDA_MLPGDT.

The comparison of multiple data sets' accuracy, precision, recall, and F-1 scores is shown in Figs. 2–4. Comparative analysis is made between the proposed SAW_WDA_MLPGDT and existing LR, LSVM, and FFNN. Based on this comparison research, the suggested technique found the virus and concept drift of the website with the best accuracy. The above results show enhanced output in web data classification and website concept drift detection.

Table 1: Comparative analysis of parameters for various datasets in terms of web data classification

Dataset	Techniques	Accuracy	Precision	Recall	F1_score
DDoS 2016	LR	97.1	90.6	89.3	86.3
	LSVM	97.3	90.8	89.5	86.5
	FFNN	97.6	91	89.8	87
	SAW_WDA_MLPGDT	98	92	90	88
UNSW-NB15	LR	96.8	90	88.1	86.5
	LSVM	97.1	90.5	88.5	87
	FFNN	97.5	91	90	87.3
	SAW_WDA_MLPGDT	97.8	92	90	88
CICIDS 2017	LR	97.3	90.5	88.5	87
	LSVM	97.5	91	88.9	87.3
	FFNN	97.8	91.8	90	87.8
	SAW_WDA_MLPGDT	98.1	94	90.5	88



(a) Accuracy (b) Precision

(c) Recall (d) F-1 score

Figure 2: Analysis based on the DDoS 2016 dataset

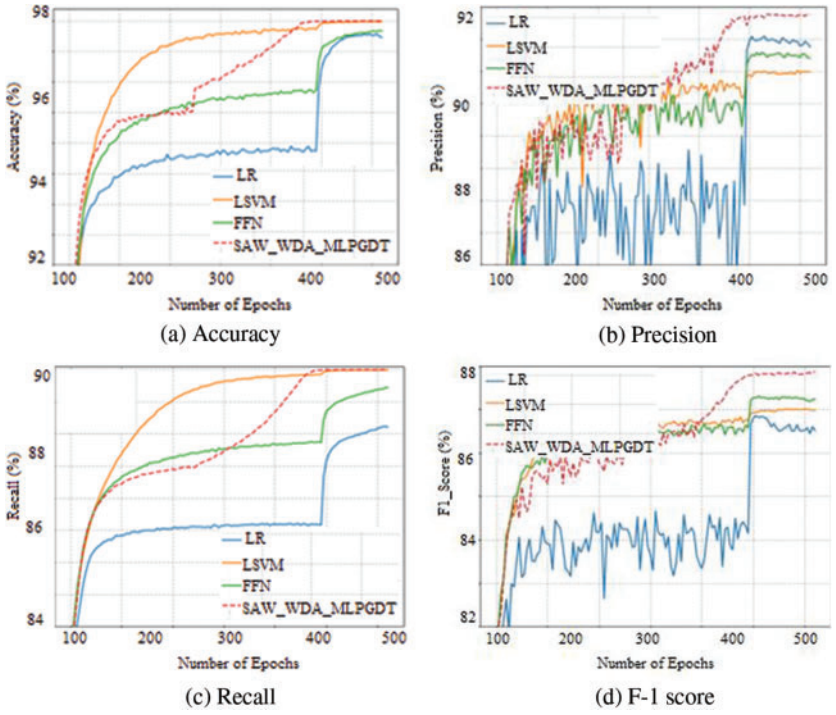


Figure 3: Analysis based on the UNSW-NB15 dataset

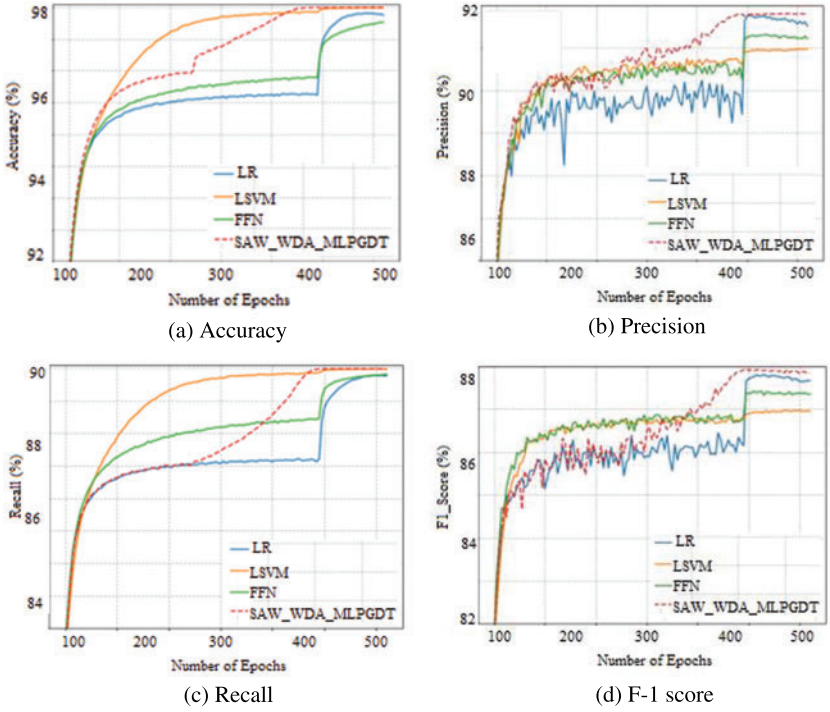


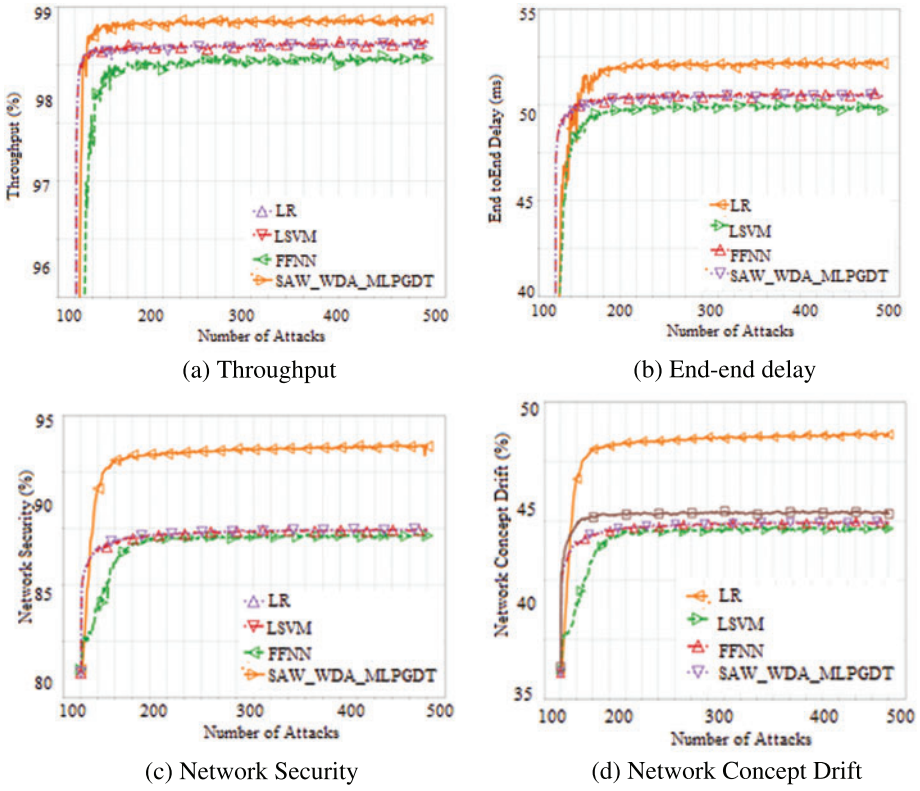
Figure 4: Analysis based on the CICIDS 2017 dataset

The proposed SAW_WDA_MLPGDT is compared to existing LR, LSVM, and FFNN with the DDoS 2016 dataset, UNSW-NB15 dataset, and CICIDS 2017 dataset. Figs. 2–4 show a comparison of the DDoS 2016 dataset, UNSW-NB15 dataset, and CICIDS 2017 dataset in terms of accuracy, precision, recall, and F-1 score.

Table 2 shows the output for the proposed technique in terms of malware and concept drift detection. After applying the proposed protocol, the data transmission rate, delay, and security have been evaluated.

Table 2: Parametric analysis for proposed and existing technique in terms of network performance

Parameters	LR	LSVM	FFNN	SAW_WDA_MLPGDT
Throughput	96.5	97	97	99
End-end delay	53.5	50	49.9	48.2
Network security	85.5	85.9	86	94.5
Network concept drift	48.5	45	44.9	43.5



(a) Throughput (b) End-end delay

(c) Network Security (d) Network Concept Drift

Figure 5: Parametric analysis of malware detection and concept drift mitigation in terms of (a) Throughput, (b) End-end delay, (c) network security, (d) network concept drift

Fig. 5 shows a parametric analysis of malware detection and concept drift comparison between existing and proposed techniques. This comparison research showed that the suggested method produced the best results for reducing concept drift on the website. This comparison research showed

that the suggested method produced the best results for reducing concept drift on the website and detect malware DDOS attacks in the network. Here the data transmission efficiency has been calculated in terms of throughput and minimal end-end delay. Malware attack mitigation has been calculated by network security in which the proposed technique has enhanced the security of the network and concept drift analysis has been made after establishing the proposed protocol.

5 Conclusion

This paper proposed a novel technique for designing the architecture in concept drift analysis and detection of malware attacks like DDOS in the network. The network has been analyzed for detecting the intrusion and concept drift using secure adaptive windowing with website data authentication protocol (SAW_WDA) integrated with authentication protocol to avoid DDOS attacks and concept drift. The user data of the network will be collected and classified using multilayer perceptron gradient decision tree (MLPGDT) classifiers. Based on the classification output, the decision for the detection of attackers and authorized users will be identified. The experimental results show output based on intrusion detection and concept drift analysis system in terms of throughput, end-end delay, network security, network concept drift, and results based on the classification in terms of accuracy, memory, and precision, and F-1 score.

Funding Statement: The Taif University Deanship of Scientific Research supported this endeavor (Project Number: 1-443-4), for which the authors are grateful to Taif University for their kind support.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] S. Singhal, U. Chawla and R. Shorey, "Machine learning & concept drift based approach for malicious Website detection," in *2020 Int. Conf. on Communication Systems & Networks (COMSNETS)*, location-Bangalore, India, pp. 582–585, 2020.

[2] D. Pradheep, R. Gokul, V. Naveen and J. Vijayarani, "Anomaly intrusion detection based on concept drift," *Global Journal of Computer Science and Technology*, vol. 20, no. 2, pp. 14–22, 2020.

[3] L. Yang, W. Guo, Q. Hao, A. Ciptadi, A. Ahmadzadeh *et al.,* "CADE: Detecting and explaining concept drift samples for security applications," in *Proc. 30th USENIX Security Symp.*, Anaheim, CA, USA, pp. 2327–2344, 2021.

[4] H. Alsuwat, E. Alsuwat, M. Valtorta, J. Rose and C. Farkas, "Modeling concept drift in the context of discrete Bayesian networks," in *11th Int. Conf. on Knowledge Discovery and Information Retrieval (KDIR 2019)*, Portugal, Polytechnic Institute of Setubal/INSTICC, pp. 214–224, 2019.

[5] D. Jayaratne, D. D. Silva, D. Alahakoon and X. Yu, "Continuous detection of concept drift in industrial cyber-physical systems using closed-loop incremental machine learning," *Discover Artificial Intelligence*, vol. 1, no. 1, pp. 1–13, 2021.

[6] J. Gama, I. Zliobaite, A. Bifet, M. Pechenizkiy and A. Bouchachia, "A survey on concept drift adaptation," *ACM Computing Surveys (CSUR)*, vol. 46, no. 4, pp. 44:1–44:37, 2014.

[7] I. Kumpulainen, "Adapting to concept drift in malware detection," Ph.D. dissertation, Aalto University, Finland, 2020.

[8] F. Ceschin, H. M. Gomes, M. Botacin, A. Bifet, B. Pfahringer *et al.,* "Machine learning (in) security: A stream of problems," arXiv preprint arXiv:2010.16045, 2010.

[9] D. Mulimani, S. G. Totad, P. Patil and S. V. Seeri, "Adaptive ensemble learning with concept drift detection for intrusion detection," *Data Engineering and Intelligent Computing*, vol. 1, no. 1, pp. 331–339, 2021.

[10] Y. Sun, Z. Wang, H. Liu, C. Du and J. Yuan, "Online ensemble using adaptive windowing for data streams with concept drift," *International Journal of Distributed Sensor Networks*, vol. 12, no. 5, Art. no. 4218973, pp. 1–9, 2016.
[11] Q. Chang and A. Sebghati, "Efficient algorithm based on adaptive window-model predictive control for automatic stacking in warehouse center," *IEEE Access*, vol. 9, no. 1, pp. 94813–94825, 2021.
[12] G. Jo, K. Jung and S. Park, "An adaptive window size selection method for differentially private data publishing over infinite trajectory stream," *Journal of Advanced Transportation*, vol. 2018, no. 12, pp. 1–11, 2018.
[13] Q. Li, Z. Wu, Z. Wen and B. He, "Privacy-preserving gradient boosting decision trees," in *Proc. of the AAAI Conf. on Artificial Intelligence*, New York, USA, pp. 784–791, 2020.
[14] M. C. Popescu, V. E. Balas, L. Perescu-Popescu and N. Mastorakis, "Multilayer perceptron and neural networks," *WSEAS Transactions on Circuits and Systems*, vol. 8, no. 7, pp. 579–588, 2009.
[15] J. Feng, Y. Yu and Z. H. Zhou, "Multilayer perceptron and neural networks," *Advances in Neural Information Processing Systems*, vol. 31, no. 1, pp. 3551–3561, 2018.
[16] H. Alla, L. Moumoun and Y. Balouki, "A multilayer perceptron neural network with selective-data training for flight arrival delay prediction," *Scientific Programming*, vol. 2021, no. 1, pp. 1–12, 2021.
[17] M. Mohammadpourfard, Y. Weng, M. Pechenizkiy, M. Tajdinian and B. Mohammadi-Ivatloo, "Ensuring cybersecurity of smart grid against data integrity attacks under concept drift," *International Journal of Electrical Power & Energy Systems*, vol. 119, no. 1, Art. no. 105947, pp. 1–13, 2020.
[18] T. H. M. Le, B. Sabir and M. A. Babar, "Automated software vulnerability assessment with concept drift," in *2019 IEEE/ACM 16th Int. Conf. on Mining Software Repositories (MSR)*, Montreal, Canada, pp. 371–382, 2019.
[19] G. Andresini, F. Pendlebury, F. Pierazzi, C. Loglisci, A. Appice *et al.,* "INSOMNIA: Towards concept-drift robustness in network intrusion detection," in *Proc. of the 14th ACM Workshop on Artificial Intelligence and Security*, Virtual Event Republic of Korea, pp. 111–122, 2021.
[20] R. Xu, Y. Cheng, Z. Liu, Y. Xie and Y. Yang, "Improved long short-term memory based anomaly detection with concept drift adaptive method for supporting IoT services," *Future Generation Computer Systems*, vol. 112, no. 1, pp. 228–242, 2020.