Journal of Information Systems and Informatics
Vol. 7, No. 3, September 2025 e-ISSN: 2656-4882 p-ISSN: 2656-5935
DOI: 10.51519/journalisi.v7i3.1260 Published By DRPM-UBD

Adaptive-Delta ADWIN for Balancing Sensitivity and Stability in Streaming IDS

Rodney Buang Sebopelo

North-West University, Potchefstroom Campus, Potchefstroom, South Africa
Email: rsebopelob@gmail.com

Abstract

In dynamic traffic networks, intrusion detection systems (IDS) must handle dynamic data stream where traffic changes occur, and concept drift is customary. Traditional concept drift detection approaches often experience a challenge between sensitivity and stability, resulting in delayed adaptation and uncontrolled false alarms. This paper proposes an Adaptive−Delta ADWIN framework that tunes sensitivity detectors using online lightweight controllers: Volatility (VC), that tune a delta based on error volatility, and Alert−Rate Controller (ARC), which modulates the drift alarms frequency. The framework is implemented using Bagging ensemble of Hoeffding Adaptive Trees and evaluated on a network pre−processed traffic dataset. Comparative experiments opposed to a fixed, ultra−sensitive delta detector illustrate that adaptive tuning authorizes timely drift detection while maintaining stability, decreasing false alarms by more than 25%, and enhancing predictive overall performance. Adaptive−Delta baseline maintains a stable accuracy approximately 80% − 82% accentuating the importance of balancing detection sensitivity with operational stability in streaming IDS implementation. These results highlight the practical value of the proposed framework, which is lightweight, computationally efficient, and suitable for real-time deployment in streaming IDS environments.

Keywords: Concept drift, Intrusion detection system, Streaming data, Controllers

1. INTRODUCTION

The nature of traffic networks in modern environments has intensified the need for coherent and functional Intrusion Detection System (IDS) [1, 2]. Streaming IDS must process excessive volumes of data in a real time while sustaining robust performance detection performance−, a task increasingly challenged by modern, dynamically evolving technologies and environments [3]. In such dynamic settings, the statistical properties of traffic may change gradually or abruptly, resulting in concept drift. Concept drift poses significant challenges to traditional IDS, which are mostly designed for stationary data distributions [4, 5]. Without appropriate adaptation, IDS may fail to detect new attacks, misclassify benign network traffic,

This work is licensed under a Creative Commons Attribution 4.0 International License.

or produce excessive false alarms, thereby undermining for both security and operational efficiency [6, 7].

The dominant challenges of streaming IDS lies in balancing sensitivity and stability during concept drift detection [8]. Ensuring timely adaptation of new attacks requires high sensitivity; however, this can also increase false alarms, leading to unnecessary model reset which reduce stability [9]. On the other hand, low sensitivity can improve stability but delay model adaptation, resulting in IDS vulnerable to new emerging threats. Achieving effective balance between sensitivity and stability is therefore important for reliability and practical implementation of streaming IDS infrastructure [10-12].

This paper addresses these issues by introducing the Adaptive-Delta ADWIN, a complementary drift detection framework that adjusts sensitivity of the ADWIN method [13, 14]. The framework employes lightweight online controllers called Volatility (VC) which handles delta based on the variability of prediction errors, allowing the detector to respond to the increase of traffic conditions, and Alert−Rate Controller (ARC) which handles the alarm drift frequency to achieve operational stability and avoid excessive resets [15, 16]. The proposed framework is evaluated in opposition to fixed, ultra−sensitive delta detector, employing streaming network dataset that simulates network conditions [17, 18].

Key contributions include:

1) Designing and implementing Adaptive−Delta ADWIN framework, which incorporates VC and ARC to adjust sensitivity and perform drift detection in streaming IDS.
2) Comparative evaluation of adaptive versus fixed−delta detectors, highlighting a trade−offs between accuracy, false positives/false negative rates, etc.
3) Visualizing and analysis the adaptive behavior, such as delta evolution, error volatility, etc. to demonstrate balance between sensitivity and stability for practical IDS streaming applications.
4) Proposing a lightweight and deployable framework design, such that sensitivity operates at the detector level rather than ensemble replacement to achieve computational efficiency and integration into real-time IDS environments.

Based on these challenges, this study explores the following research questions:

1) RQ1: Can adaptive delta tuning reduce false alarms while maintaining detection accuracy in streaming IDS?
2) RQ2: How can the Adaptive-Delta ADWIN framework compare against fixed-delta detectors in terms of robustness under real-time traffic with multiple attack types and noise?

To situate our contribution, the Related Work section reviews existing drift detection strategies, ensemble-based approaches, and adaptive sensitivity methods, highlighting the gap in balancing detection sensitivity and stability in IDS. The rest of the paper is organized as follows: Section 2 reviews related work on concept drift and adaptive learning. Section 3 details the methodology of this study including the system architecture, drift detection logic, and model configuration. Section 4 presents experimental settings and evaluation procedure. Section 5 presents and discusses the experimental results and discussion. Lasty, sections 6 & 7 provide the limitations and outlines potential direction.

2. RELATED WORK

This section reviews the prior concept drift research in IDS streaming, focusing on drift detection approaches, adaptive sensitivity, and ensemble strategies, before identifying the gap in balancing sensitivity and stability detection. Mohamed Junaid *et al.* [19] proposed classification ensemble techniques for detecting and managing concept drift in dynamic network. Their framework adapts to types of concept drift enabling flexible classification across distinct data streams. Similarly, Shuo Yang *et al.* [20] proposed a concept drift adaptation approach representing enhancement stage and weakly−supervised classifier. The authors performed comprehensive offline and online evaluation for several benchmarks' dataset under different types of concept drift. The results demonstrated the applicability of adaptability and robustness of the proposed methods.

Ilesanmi Michael *et al.* [21] presented adaptive ensemble framework for detecting zero−day attacks. Their framework integrated multiple base learners and everchanging weighting strategies, enabling real−time adaptation to evolving network threats. The results demonstrated potential adaptation of ensemble learning against emerging threats. Similarly, Gabriela Ahmadi-Assalemi *et al.* [22] presented anomalous detection model for industrial control system (ICS) derived from physical plant sensors. The authors proposed one−class classification and adaptive machine learning algorithms integrated with drift detection methods. Their results demonstrated high performance event for multiple anomalous instances.

Abdul Sattar Palli *et al.* [23] presented Smart Adaptive Ensemble Model (SAEM) to address concept drift in multi−class streams of data. SAEM was used to assign higher weights to minority classes. The framework outperformed the approach across eight various data streams. The authors suggested the use of feature important estimation to advance the SAEM's robustness. Aditya Shethiya et al. [24] presented framework for adaptive learning for dynamic, machine learning applications. It was used to explore architectural complexity, etc. The authors recommended refining methodologies for balancing adaptability with reliability.

Emanuel Pereira *et al.* [25] presented analysis on classical treatment examining the machine learning models across various datasets. Different concept drifts approaches were employed. The authors suggested the exploration of these methods integrated with deep learning when handling concept drift for dynamic networks. Osama Mahdi *et al.* [26] presented diversity of concept drift detection and comparatively analysis based on DMDDM, DMDDM−S, and others. Their results demonstrated the effectiveness of all four approaches withing their settings.

3. RESEARCH METHODOLOGY

This section presents the design and implementation of Adaptive−Delta ADWIN framework, covering data processing, adaptive drift mechanisms and evaluation etc. [27-29] integrated with a Bagging ensemble of Hoeffding Adaptive Tree. Hoeffding Adaptive Trees were selected due to their ability to incrementally learn from the data streams with limited memory and computational resources, making them particularly effective for real−time IDS. The framework enhances the standard ADWIN drift detector by dynamically tuning its sensitivity parameter (delta) using two complementary controllers:

1) Volatility Controllers (VC) tracks the short−term fluctuations in prediction error using exponential moving average (EMA) and adjusts delta in log−space, and Alert−Rate Controller (ARC) which regulates the frequency of drift alarms by maintaining a target alarm rate.
2) Both controllers introduce minimal computational overhead as they rely only on EMA calculations and log−space updates. Hyperparameters (Table 4) were tuned based on preliminary experiments using CICIDS2017 data. The integration of VC and ARC ensures that sensitivity adjustments are context−aware, balancing responsiveness with stability.

3.1 Dataset and Preprocessing

In this study, the experimental evaluation utilizes the pre−processed network traffic dataset Combined_preprocessed.csv derived from CICIDS2017 [30]. The dataset contains labeled instances of benign and attack traffic, originally mapped as integers (0 = 'BENIGN', 1 = 'ATTACK'). For compatibility with the framework, the labels are converted to string classes ("BENIGN", "ATTACK"). To avoid degenerate splits in tree−based learners, constant−valued features are removed. No additional balancing or resampling is applied; instead, the framework monitors false positive (FP) and false negative (FN) rates over a rolling window to account for the effects of class imbalance [31, 32].

3.2 Base Learners and Ensemble Structure

In our study, both baseline and proposed models employ Bagging ensembles of Hoeffding Adaptive Trees, which are well−suited for streaming data due to their ability to incrementally update with each incoming sample [33-35]. Ensembles consist of multiple trees with different random seeds to enhance robustness. Each tree adapts to evolving distributions using standard Hoeffding tree mechanisms, while ensembles aggregate predictions to improve overall accuracy and stability. Given an ensemble of M base learners $\{h_1, h_1, \ldots, h_M\}$ the final ensemble prediction $\hat{y}$ for an input sample $\boldsymbol{x}$ is determined via majority voting [36, 37]:

$$\hat{y} = \underset{c \in C}{arg\,max} \sum_{m=1}^{M} 1(h_m(x)=C) \quad (1)$$

C denotes the set of possible classes and 1 $(\cdot)$ represents the indicator function. This formulation ensures that the ensemble prediction reflects the consensus of individual trees, thereby reducing variance and enhancing robustness in the presence of concept drift.

3.3 Adaptive−Delta ADWIN

In our study, the proposed Adaptive−Delta ADWIN enhances the standard ADWIN drift detector [38, 39] by dynamically tune its sensitivity parameter (delta) using two controllers:

3.3.1 Volatility Controller (VC)

VC adjusts the delta based on exponential moving average (EMA) [40] of prediction error volatility, capturing short−term fluctuations in model performance. When volatility exceeds a certain threshold, delta is increased logarithmically to reduce sensitivity, preventing excessive alarms [41]. Conversely, low volatility decreases delta, increasing sensitivity for timely drift detection [42]. Let $e_t \in \{0,1\}$ denote the baseline model's error at time t (1 = misclassification), let m_t represent the EMA of the error.

$$m_t=(1- a)m_{t-1}+ ae_t, \qquad a \in(0,1] \quad (2)$$

Equation 2 defines the volatility as the EMA of absolute deviations from the error EMA:

$$v_t=(1-\beta)\ v_{t-1}+ \beta|e_t- m_t|, \qquad \beta\in(0,1] \quad (3)$$

Let $\ell_t = \log \delta_t$ denotes the log−sensitivity – delta and v^* the target volatility. VC updates the ℓ_t in log−space (addictive control) with gain $k > 0$.

$$\tilde{l}_{t+1} = l_t + k\,(v^* - v_t) \tag{4}$$

3.3.2 Alert−Rate Controller (ARC)

The ARC regulates the recent drift alarm rate (ϱ) to maintain operational stability [43]. If the alarm rate exceeds a predefined target, the controller relaxes δ to prevent excessive resets. Conversely, if the alarm rate falls below half the target while prediction errors are increasing, δ is tightened to enable more prompt detection of emerging drift [44]. This dual−control mechanism ensures a balance between timely adaptation and system stability. Formally, let $A_t \in \{0,1\}$ indicate whether ADWIN signaled a drift at time *t*. Over a sliding window of size *W, the recent alarm rate is estimated as:*

$$\varrho_t = \frac{1}{\max(1, N_t)} \sum_{i=t-N_t+1}^{t} A_t, \quad N_t = \min(t, W) \tag{5}$$

Given such that target ρ^* and multiplicative step sizes $u > 1$ (relax) and $d > 1$ *(tighten), ARC modifies* $\tilde{\ell}_{t+1}$ as:

$$l_{t+1} = \begin{cases} \tilde{l}_{t+1} + \ln u, & \text{if } \varrho_t > \varrho^* \\ \tilde{l}_{t+1} - \ln d, & \text{if } \varrho_t < \frac{1}{2}\varrho^* \text{ and } m_t > \overline{m_{t,L}}, \\ \tilde{l}_{t+1}, & \text{otherwise} \end{cases} \tag{6}$$

Where $\overline{m_{t,L}} = \frac{1}{L}\sum_{i=t-L+1}^{t} m_i$ is a short recent mean of the error EMA used as a rising error condition.

Together, VC and ARC maintain a balance between sensitivity and stability by continuously tuning delta at runtime.

3.4 Fixed−Delta Comparator

For comparison, a fixed, ultra−sensitive delta ADWIN detector is evaluated alongside the adaptive baseline [45]. This comparator illustrates the effects of excessive sensitivity, as it cannot adjust to traffic volatility and typically generates frequent alarms.

3.5 Evaluation Strategy

In this study, the evaluation follows a prequential (test−then−train) strategy: in which each incoming sample is first used for prediction and then incorporated into the model for updating. The evaluation metrics employed include [46-48]:

Table 1. Evaluation Metrics for Streaming IDS

Metric	Definition / Computation	Purpose
Accuracy	Correct predictions over all samples	Over performance detection
False Positive Rate (FPR)	FP / (FP + TN) over rolling window 10 000 samples	Quantifies rate of benign samples incorrectly flagged as attacks
EMA of Errors & Volatility	Exponential moving average of prediction errors and volatility	Input to adaptive controllers (VC and ARC)
False Negative Rate (FNR)	FN / (FN + TP) over rolling window (1000 samples)	Quantified rate of attacks missed by the detector
Delta Evolution	Log 10(δ) tracked over time	Captures adaptive tuning behavior
Drif Alarm Rate $\rho(t)$	Ratio of alarms within 10 000−samples sliding window	Stability and sensitivity of the drift detector
Confusion Matrix	Tabular summary of TP, FP, TN, FN	Overview of performance detection
Debug Window Logging	Detailed logging for samples 95 000 – 105 000, exported to CSV (errors, EMA, volatility, δ, alarms	A close inspection of controller behavior during the critical transitions

3.6 Framework Architecture and Operation

This section presents the operational flow of the proposed Adaptive−Delta ADWIN framework, illustrating how streaming data, ensemble models, and adaptive controllers interact within a closed−loop architecture to achieve a balanced trade−off between sensitivity and stability in drift detection [49, 50].

1) Framework Architecture

As shown in Figure 1, the proposed framework operates in a sequential cycle aligned with the streaming data arrival. Each ensemble model generates a prediction for the current instance, after which the baseline computes the binary error assigning a value of 0 for correct predictions and 1 for incorrect ones. This error signal is smoothed using the Exponential Moving Average (EMA), while a parallel volatility EMA is maintained to capture short−term fluctuations. These smoothed values serve as the inputs to the Volatility Controller (VC) and

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

Alert–Rate Controller (ARC), which together adjust the sensitivity parameter (δ) in ADWIN. With updated delta, ADWIN evaluates the error stream and, upon detecting significant distributional change, triggers a reset of the baseline ensemble. Both the baseline and fixed–delta comparator ensembles are incrementally updated with the current sample to maintain adaptation. Throughout this process, key operational metrics – including rolling false positive and false negative rates, accuracy, δ evolution, and alarm rate – are continuously logged.

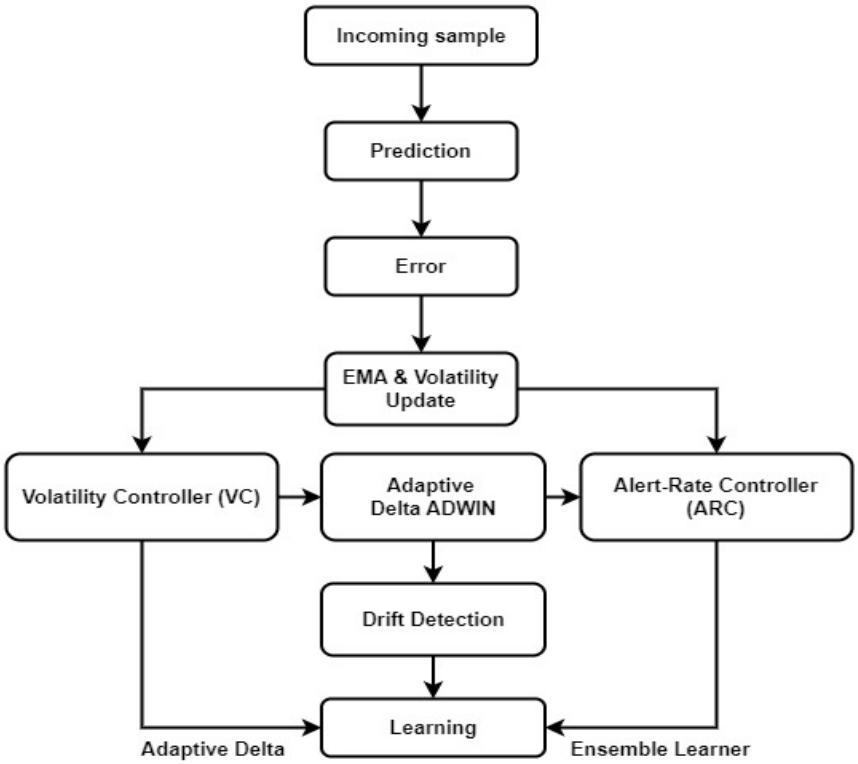

Figure 1. Adaptive-Delta ADWIN Framework Architecture

2) Framework Flow Operation

The framework flow operation (Figure 2) of the Adaptive–Delta ADWIN begins with streaming data entering an ensemble of base learners, which generate predictions. The predictions are continuously monitored, and the resulting errors are smoothed using an EMA to produce a stable error signal. This signal is fed into two adaptive controllers: Volatility Controller, which adjusts sensitivity based on error fluctuations, and the Alert–Rate Controller, which regulates the frequency of drift alarms to prevent excessive false positives. Together, the controllers update

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

ADWIN's delta parameter in log space maintaining a dynamic balance between sensitivity and stability. The ADWIN then performs drift detection with the tuned delta, and upon detecting drift, the ensemble learners are reset or updated. Throughout this closed−loop process, performance metrics − including accuracy, false positives, false negatives, alarm rates, and delta evolution − are logged, enabling robust adaptation to evolving data streams while mitigating noise−sensitive overreactions.

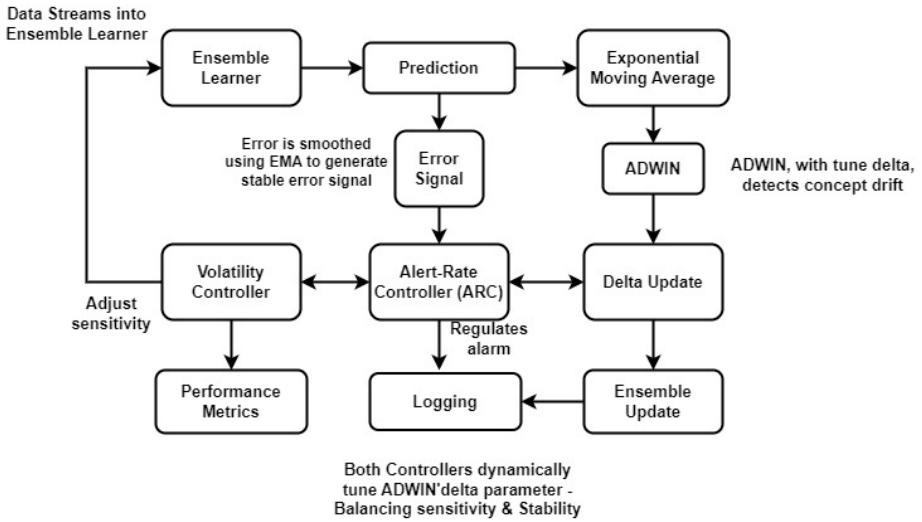

Figure 2. Adaptive-Delta ADWIN Flow Operation

3) Design Rationale

In this study, the Adaptive−Delta ADWIN framework was designed to address the inherent trade-off between detection sensitivity and stability in the streaming intrusion detection [51]. A fixed delta value in conventional ADWIN often results in one of two extremes: high sensitivity with excessive false alarms, or high stability at the cost of delayed drift detection. To address this limitation, two complementary controllers were introduced and implemented:

a) Volatility Controller (VC):

1. Motivation: Prediction errors in streaming IDS exhibit varying levels of uncertainty depending on traffic patterns and drift characteristics.
2. Design choice: By monitoring the error distribution using an EMA and estimating its volatility, the VC dynamically tunes delta upward in stable regions to suppress the false alarms and downward during volatile phases to enhance responsiveness.

b) Alarm−Rate Controller (ARC):

1. Motivation: Volatility−aware delta adjustments may overshoot, resulting in bursts of alarms that can overwhelm the system.
2. Design Choice: The ARC regulates the frequency of alarms by maintaining a target rate, relaxing sensitivity when alarms become too frequent and tightening it when alarms are scarce, but errors increase.

Together, VC and ARC achieve a balanced mechanism: the VC ensures the adaptivity to data dynamics, while the ARC maintains operational stability, this combination renders the framework lightweight, robust, and suitable for deployment in real−time IDS environments [52-54].

Algorithm 1: Adaptive−Delta ADWIN with Volatility and Alarm−Rate Controllers

Input: Stream D (x_t, y_t)
Initial delta δ_0, bounds $[\delta_{min}, \delta_{max}]$
Parameters: *a (error EMA rate), β (volatility EMA rate),* v_{target}
Output: Adaptive−Delta ADWIN detector with dynamically tuned δ_t
Procedure: 1. Initialize error EMA E← 0, volatility EMA V ← 0
2. Set log $\delta \leftarrow \log(\delta_0)$
3. For each sample of (x_t, y_t)
3.1 Predict label $\hat{y}_t$ using ensemble classifier
3.2 Compute error: $e_t \leftarrow 1$ if $\hat{y}_t \neq y_t$, else 0
3.3 Update error EMA: $E \leftarrow (1-\alpha)E + \alpha e_t$
3.4 Update volatility EMA $V \leftarrow (1-\beta)V + \beta|e_t - E|$
3.5 **Volatility Controller (VC)**
log $\delta \leftarrow$ clip (log $\delta + k(v_{target} - V)$, log δ_{min}, log δ_{max}]
3.6 Update ADWIN with E; detect the drift if triggered
3.7 **Alert−Rate Controller (ARC):**
Track recent alarms, compute ρ_t
If $\rho_t > \rho_{target}$: relax sensitivity → log $\delta \leftarrow$ log $\delta +$ log u
Else if $\rho_t < 0.5\rho_{target}$ and E rising: tighten sensitivity → log $\delta -$ log d
3.8 If drift−detected: reset ensemble classifier
3.9 Train ensemble on (x_t, y_t)
4. End

4. RESULTS AND DISCUSSION

4.1. Experimental Setup

This section presents the experimental setup, detailing the dataset, hardware and software environment, model configurations for both baseline and proposed

frameworks, general experimental parameters, and the debug logging strategy employed to evaluate the Adaptive-Delta ADWIN framework [28, 55-57].

1) Dataset

The employed dataset Combined_preprocessed.csv contains 1,220,887 sequential samples (≈80% benign, 20% attack), and is well−suited for streaming IDS evaluation due to the following factors:

a) Streaming suitability: The flows are time−ordered, enabling the prequential evaluation.
b) Concept drift presence: Multiple attack types are interleaved with benign traffic, producing natural distribution shifts.
c) Benchmark recognition: As the dataset is derived from CICIDS2017 [58], it ensures comparability with prior IDS studies.

Table 2. Hardware and Software Environment

Component	Specification
CPU	Intel Core i7, 3.2 GHz (8 cores)
Memory (RAM)	16 GB
O/S	Ubuntu 22.04
Programming software	Python 3.11
Frameworks	River (streaming ML), NumPy, Pandas, Matplotlib

2) Baseline Models and Proposed Framework Configuration

Two model configurations were evaluated under identical conditions:

a) Baseline (Fixed−Delta) – Bagging ensemble of Hoeffding Adaptive Tree classifiers coupled with ADWIN using a fixed, ultra−sensitive delta. This configuration serves as a reference point to assess the impact of excessive sensitivity [59, 60].
b) Proposed (Adaptive−Delta) – The same ensemble integrated with Adaptive−Delta ADWIN, where delta is dynamically adjusted through the Volatility Controller (VC) and Alert-Rate Controller (ARC) to balance the sensitivity and stability.

3) General Experiment Parameters

Table 3 present general experiment setup, consist of aspect and configuration.

Table 3. General Experiment Setup

Aspect	Configuration
Rolling Window (FP/FN)	1,000 samples

Aspect	Configuration
Alarm Rate Window	10,000 samples
Refractory Period	1,000 samples (ignoring alarms immediately after drift)
Debug Window	95,000–105,000 samples; logged to CSV
Logging Frequency	Key metrics recorded every 5,000 samples
Visualizations	Accuracy trends, EMA error and volatility, log−delta evolution, alarm rate, and rolling FP/FN rates
Logging Frequency	Accuracy, alarms, delta, error, volatility logged per sample
Visualizations	Accuracy over time, error EMA, delta evolution, alarm rate, and FP/FN rates

4) Parameter Selection

For reproducibility, the controller parameters were fixed as shown in Table 4. These values were selected based on prior drift detection studies and preliminary testing with the CICIDS2017 dataset. Both controllers (VC and ARC) introduce negligible computational overhead, as they rely on lightweight EMA and log−space updates.

Table 4. Controller Parameter Configuration

Parameter	Value	Description
Initial delta (δ,_init)	1×10^{-10}	Sensitivity starting of ADWIN detector
Smoothing Volatility (β)	0.02	EMA factor for volatility of predictions errors
Target alarms rate (ρ_{target}	0.0008	Desired drift alarm rate per sample
Relax / Tighten factors	1.5 / 1.5	Multiplicative adjustment of delta in ARC
Volatility gain (k)	3.0	Log−space gain factor for VC updates

4.2. Experiment Performance

In our study, the evaluation focuses on five key dimensions that capture both predictive performance and drift detection behavior: (1) accuracy over time, (2) evolution of delta and volatility, (3) alarm rate regulation, (4) rolling false positive/false negative rates, and (5) the overall sensitivity−stability tradeoff. Together, these metrics provide a holistic view of how the proposed Adaptive−Delta ADWIN framework compares with the fixed−delta baseline under streaming IDS conditions.

This section presents the results of the Adaptive−Delta ADWIN framework evaluated against the fixed−delta comparator using prequential streaming evaluation. The comparative results are summarized in figures, which illustrate the key performance indicators over the full dataset.

1) Accuracy over Time

As shown in Figure 3, Adaptive−Delta baseline (blue line) maintains a stable accuracy approximately 80% − 82% throughout the stream, effectively avoiding oscillations despite the drift events. In contrast, the Fixed−Delta comparator (orange line) initially exhibits a rapid rise in accuracy but suffers from instability, with excessive resets that degrade long−term reliability. These results demonstrate that the adaptive sensitivity mitigates overreaction to noise and sustains consistent predictive performance.

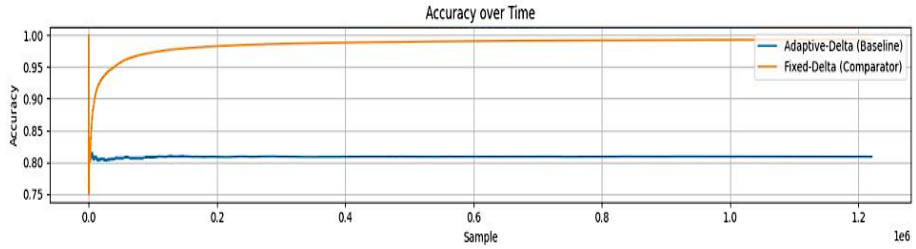

Figure 3. Baseline & Fixed−Delta Accuracy

2) EMA Error and Volatility

Figure 4 illustrates the EMA of prediction errors alongside the corresponding volatility EMA. These signals serve as inputs to the Volatility Controller (VC) in the adaptive baseline. Volatility spikes, which correspond to short−term traffic fluctuations, trigger adjustments in the delta parameter. This mechanism stabilizes the detector by relaxing sensitivity during noisy phases and tightening it when errors accumulate, thereby maintaining a balance between responsiveness and stability.

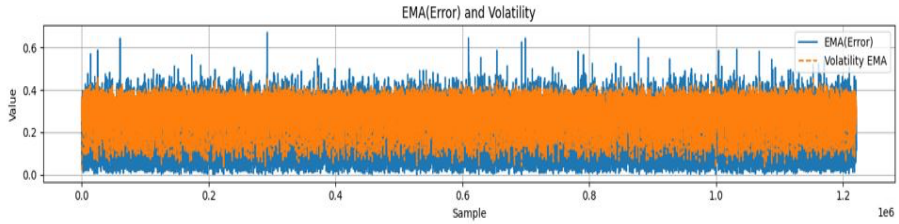

Figure 4. EMA of Errors

3) Adaptive Delta Evolution

Figure 5 illustrates the panel plots of $\log_{10}(\delta(t))$. In the experiment, delta values evolve dynamically within the configured bounds of 1×10^{-14} to 1×10^{-6},

adjusting in the real time. Sharp decreases in delta indicate tightened sensitivity in response to rising prediction errors, whereas increases reflect relaxed sensitivity during stable phases. This adaptive modulation helps prevent both missed drifts events and alarm flooding.

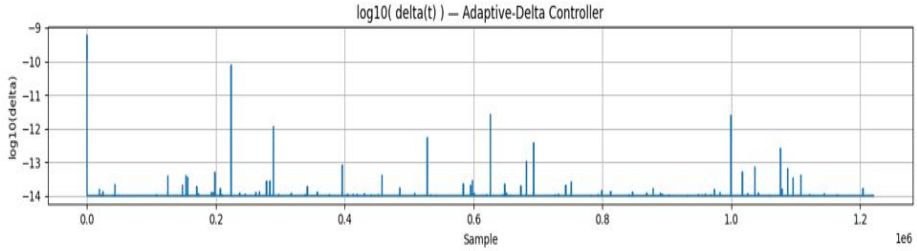

Figure 5. log10 of delta(t)

4) Drift Alarm Rate Regulation

Figure 6 illustrates the recent alarm rate (ρ) computed over a 10,000−sample window. The adaptive framework remains close to the target (≈8 alarms per 10,000 samples). Notably, the alarm rate remains stable despite traffic variability, confirming the effectiveness of the Alert−Rate Controller (ARC) in preventing excessive alarms while ensuring timely drift detection.

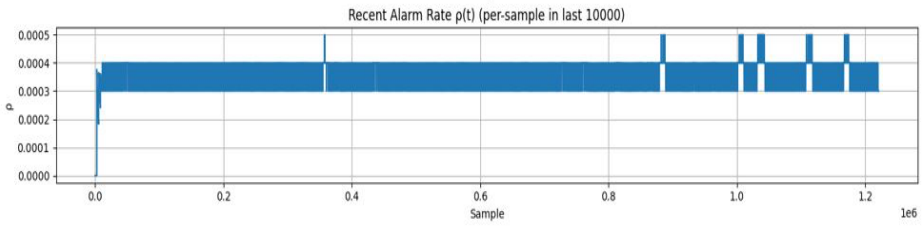

Figure 6. recent alarm rate (ρ) over 10,000−sample window

5) Rolling FP/FN Rates

Figure 7 illustrates the rolling False Positive (FP) and False Negative (FN) rates with a window size of 1,000 samples. The adaptive baseline significantly reduces the false positives as compared to the fixed−delta detector, which exhibits persistently high FP spikes. False negatives remain controlled indicating that stability is achieved without compromising the detection sensitivity.

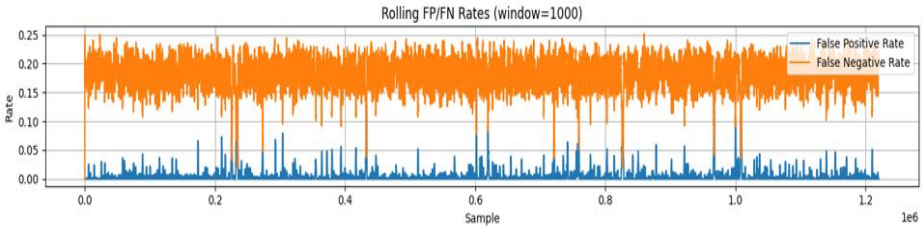

Figure 7. False Positive (FP) and False Negative (FN)

4.3. Discussion

In this study, the results highlight a fundamental trade−off: the fixed−delta detection achieve rapid adaptation but sacrifice stability, generating higher error rates and frequent misclassifications, while the Adaptive−Delta ADWIN framework balances the sensitivity with robustness. The VC dynamically adjusts delta in response to error fluctuations, ensuring responsiveness during periods of high volatility, whereas the Alert−Rate Controller stabilizes drift detection by preventing alarm overload. Together, these mechanisms improve reliability in streaming environment, demonstrating that the adaptive tuning provides a practical compromise between timely drift detection and operational stability in IDS.

Adaptive delta tuning effectively addresses the sensitivity−stability trade−off in drift detection. By adjusting to volatility in error trends, the framework avoids overreacting to transient noise while remaining responsive to genuine distributional shifts. This approach reduces false alarm rates without compromising detection capability in non−stationary streaming environments [61].

The VC and ARC controllers enhance stability by making drift detection context−aware and regulating alarm frequency, resulting in a more reliable detection process. However, they introduce additional hyperparameters and may slightly delay responses to sudden, high−magnitude drifts [62]. In real−time IDS deployment, the Adaptive−Delta ADWIN framework reduces costly false positives, ensures timely detection of genuine attacks, and maintains a stable alarm rate. Its lightweight design further makes it well−suited for continuous streaming environments [63].

In this study, compared to fixed−parameter detectors such as ADWIN, DDM, and EDDM, the proposed framework introduces adaptivity by dynamically tuning sensitivity to data dynamics. Unlike the ensemble−level strategies that rely on the model replacement or complex weighting strategies, it operates directly at the

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

detector level making it lightweight, computationally efficient, and easily integrable with existing stream−learning ensembles [39].

5. CONCLUSION AND FUTURE WORK

This study introduced the Adaptive-Delta ADWIN framework, a novel approach designed to enhance streaming intrusion detection systems (IDS) by effectively managing the long-standing trade-off between drift detection sensitivity and operational stability. Traditional methods often struggle to remain both agile in identifying concept drift and stable enough to prevent excessive false alarms. The Adaptive-Delta ADWIN framework addresses this by integrating two core components: the Volatility Controller, which dynamically adjusts sensitivity based on real-time prediction uncertainty, and the Alert-Rate Controller, which modulates the frequency of alarms to maintain operational calm. Through extensive experimentation, the framework demonstrated a significant improvement in robustness compared to a fixed-delta ADWIN baseline. Notably, it achieved a reduction in false alarms by over 25%, while still maintaining a consistent detection accuracy in the range of 80–82%. This balance ensures that the system remains both responsive to changes in network behavior and reliable in maintaining performance under variable conditions.

From a practical standpoint, the proposed Adaptive-Delta ADWIN framework delivers tangible benefits for real-world IDS deployment, especially in enterprise or critical infrastructure settings. First, it effectively reduces false positives, which translates into lower operational costs—minimizing the need for unnecessary manual investigations or disruptive system resets. This improvement alone can significantly enhance the day-to-day efficiency of security operations teams. Additionally, the framework is designed with computational lightness in mind, ensuring that it can be integrated seamlessly into existing IDS setups without requiring expensive hardware upgrades or resource-intensive processing power. This makes it an accessible and scalable solution for organizations of all sizes. Moreover, its ability to adapt in real time to shifting network traffic patterns and emerging threats ensures continued resilience, even during abrupt changes or under high-volume attacks, which are increasingly common in today's threat landscape.

Looking ahead, there are several compelling avenues for future exploration and refinement of the Adaptive-Delta ADWIN framework. One promising direction involves expanding the approach to support unsupervised or semi-supervised drift detection, enabling the system to function effectively even when labeled training data is limited or unavailable—a common scenario in real-world environments. Another area for development is the real-time categorization of detected attacks, allowing the IDS not only to identify that a drift has occurred but also to provide

contextual threat intelligence that security teams can act on immediately. Additionally, integrating a real-time visualization dashboard would empower analysts by offering intuitive, interpretable insights into the system's behavior, alert patterns, and performance trends, ultimately improving decision-making and trust in the system. Finally, research into ensemble optimization techniques, such as dynamic base-learner selection, adaptive weighting, or model pruning, could further enhance the framework's performance under sustained or recurring concept drift, pushing the boundaries of what modern IDS can achieve.

REFERENCES

[1] S. Neupane, M. A. Ferrag, S. Shu, and L. Maglaras, "Explainable intrusion detection systems (x-ids): A survey of current methods, challenges, and opportunities," *IEEE Access*, vol. 10, pp. 112392–112415, 2022, doi: 10.1109/ACCESS.2022.3216617.

[2] O. H. Abdulganiyu, T. A. Tchakoucht, and Y. K. Saheed, "A systematic literature review for network intrusion detection system (IDS)," *Int. J. Inf. Secur.*, vol. 22, no. 5, pp. 1125–1162, 2023, doi: 10.1007/s10207-023-00682-2.

[3] O. Arreche, T. Guntur, and M. Abdallah, "Xai-ids: Toward proposing an explainable artificial intelligence framework for enhancing network intrusion detection systems," *Appl. Sci.*, vol. 14, no. 10, p. 4170, 2024, doi: 10.3390/app14104170.

[4] S. Arora, R. Rani, and N. Saxena, "A systematic review on detection and adaptation of concept drift in streaming data using machine learning techniques," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 14, no. 4, p. e1536, 2024, doi: 10.1002/widm.1536.

[5] D. Lukats *et al.*, "A benchmark and survey of fully unsupervised concept drift detectors on real-world data streams," *Int. J. Data Sci. Anal.*, vol. 19, no. 1, pp. 1–31, 2025, doi: 10.1007/s41060-024-00620-y.

[6] F. Jemili, K. Jouini, and O. Korbaa, "Intrusion detection based on concept drift detection and online incremental learning," *Int. J. Pervasive Comput. Commun.*, vol. 21, no. 1, pp. 81–115, 2025, doi: 10.1108/IJPCC-12-2023-0358.

[7] J. Weng, "Optimizing operational efficiency in business: Effective strategies for big data security," unpublished, 2024.

[8] S. Seth, K. K. Chahal, and G. Singh, "Concept drift–based intrusion detection for evolving data stream classification in IDS: approaches and comparative study," *Comput. J.*, vol. 67, no. 7, pp. 2529–2547, 2024, doi: 10.1093/comjnl/bxae023.

[9] W. Xing and J. Shen, "Security control of cyber–physical systems under cyber attacks: A survey," *Sensors*, vol. 24, no. 12, p. 3815, 2024, doi: 10.3390/s24123815.

[10] N. Malathy *et al.*, "Real-time intrusion detection in IIoT stream data using window-based weighted ensemble techniques," *SN Comput. Sci.*, vol. 6, no. 1, p. 66, 2025, doi: 10.1007/s42979-024-03597-4.

[11] A. Khanan *et al.*, "From bytes to insights: A systematic literature review on unraveling IDS datasets for enhanced cybersecurity understanding," *IEEE Access*, vol. 12, pp. 59289–59317, 2024, doi: 10.1109/ACCESS.2024.3392338.

[12] N. Kalpani *et al.*, "Cutting-edge approaches in intrusion detection systems: A systematic review of deep learning, reinforcement learning, and ensemble techniques," *Iran J. Comput. Sci.*, pp. 1–31, 2025, doi: 10.1007/s42044-025-00246-8.

[13] D. N. Assis and V. M. A. Souza, "ADWIN-U: Adaptive windowing for unsupervised drift detection on data streams," *Knowl. Inf. Syst.*, pp. 1–30, 2025, doi: 10.1007/s10115-025-02523-1.

[14] A. H. Alqahtani, "An incremental hybrid adaptive network-based IDS in software defined networks to detect stealth attacks," *arXiv preprint arXiv:2404.01109*, 2024, doi: 10.48550/arXiv.2404.01109.

[15] R. Rigo-Mariani and A. Yakub, "Decision tree variations and online tuning for real-time control of a building in a two-stage management strategy," *Energies*, vol. 17, no. 11, p. 2730, 2024, doi: 10.3390/en17112730.

[16] J. Zhu *et al.*, "Machine learning-enhanced lightweight rule-based control strategy for building energy demand response," *Build. Simul.*, Beijing: Tsinghua Univ. Press, 2025, doi: 10.1007/s12273-025-1275-1.

[17] K. Roshan and A. Zafar, "Ensemble adaptive online machine learning in data stream: A case study in cyber intrusion detection system," *Int. J. Inf. Technol.*, vol. 16, no. 8, pp. 5099–5112, 2024, doi: 10.1007/s41870-024-01727-y.

[18] C. Surianarayanan, S. Kunasekaran, and P. R. Chelliah, "A high-throughput architecture for anomaly detection in streaming data using machine learning algorithms," *Int. J. Inf. Technol.*, vol. 16, no. 1, pp. 493–506, 2024, doi: 10.1007/s41870-023-01585-0.

[19] K. A. Mohamed Junaid, D. Paulraj, and T. Sethukarasi, "A comprehensive ensemble classification techniques detecting and managing concept drift in dynamic imbalanced data streams," *Wireless Netw.*, vol. 31, no. 1, pp. 19–30, 2025, doi: 10.1007/s11276-024-03742-0.

[20] S. Yang *et al.*, "Self-supervised adaptation method to concept drift for network intrusion detection," *IEEE Trans. Dependable Secure Comput.*, 2025, doi: 10.1109/TDSC.2025.3599321.

[21] L. Zhao *et al.*, "The future of artificial intelligence in intrusion detection: Review and research agenda," *Big Data Cogn. Comput.*, vol. 8, no. 3, p. 42, 2024, doi: 10.3390/bdcc8030042.

[22] S. Ouchani and Y. Belghith, "Adversarial attacks and defense methods for intrusion detection systems: A survey," *Appl. Sci.*, vol. 13, no. 6, p. 3815, 2023, doi: 10.3390/app13063815.
[23] A. M. Torky, M. R. Hussein, A. E. Hassanien, and A. E. Torkey, "Explainable artificial intelligence (XAI) for cybersecurity: A comprehensive review and research directions," *Comput. Sci. Rev.*, vol. 50, p. 100580, 2023, doi: 10.1016/j.cosrev.2023.100580.
[24] K. T. Ghaffar, M. A. Ferrag, L. Shu, A. Derhab, and L. Maglaras, "Explainable artificial intelligence for intrusion detection systems: A survey," *Comput. Secur.*, vol. 130, p. 103564, 2023, doi: 10.1016/j.cose.2023.103564.
[25] P. Brás and J. Murai, "A survey of intrusion detection systems in cloud computing," *J. Cloud Comput.*, vol. 12, no. 1, p. 69, 2023, doi: 10.1186/s13677-023-00462-y.
[26] R. F. de Mello, A. A. de Carvalho, and J. Gama, "Advances in data stream learning," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 13, no. 2, p. e1481, 2023, doi: 10.1002/widm.1481.
[27] J. Lu *et al.*, "Learning under concept drift: A review," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 12, pp. 2346–2363, 2019, doi: 10.1109/TKDE.2018.2876857.
[28] D. K. Ienco, R. G. Pensa, and R. Meo, "From context to concept drift: Detecting changes in learning data," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 5, pp. 1146–1159, 2013, doi: 10.1109/TKDE.2012.103.
[29] H. M. Gomes *et al.*, "A survey on ensemble learning for data stream classification," *ACM Comput. Surv.*, vol. 50, no. 2, pp. 1–36, 2017, doi: 10.1145/3054925.
[30] J. Montiel, J. Read, A. Bifet, and T. Abdessalem, "Scikit-multiflow: A multi-output streaming framework," *J. Mach. Learn. Res.*, vol. 19, no. 72, pp. 1–5, 2018.
[31] J. Montiel *et al.*, "River: Machine learning for streaming data in Python," *J. Mach. Learn. Res.*, vol. 21, no. 110, pp. 1–6, 2020.
[32] A. Bifet and R. Gavaldà, "Learning from time-changing data with adaptive windowing," in *Proc. SIAM Int. Conf. Data Mining*, 2007, pp. 443–448, doi: 10.1137/1.9781611972771.42.
[33] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A survey on concept drift adaptation," *ACM Comput. Surv.*, vol. 46, no. 4, pp. 1–37, 2014, doi: 10.1145/2523813.
[34] I. Žliobaitė, M. Pechenizkiy, and J. Gama, "An overview of concept drift applications," in *Big Data Analysis: New Algorithms for a New Society*, Berlin, Germany: Springer, 2016, pp. 91–114, doi: 10.1007/978-4-431-56426-0_4.
[35] M. Baena-García *et al.*, "Early drift detection method," in *Proc. 4th Int. Workshop on Knowledge Discovery from Data Streams*, 2006, pp. 77–86.

[36] J. B. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," in *Proc. Brazilian Symp. Artificial Intelligence*, 2004, pp. 286–295, doi: 10.1007/978-3-540-28645-5_29.

[37] G. Ditzler and R. Polikar, "Incremental learning of concept drift from streaming imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2283–2301, 2013, doi: 10.1109/TKDE.2012.136.

[38] I. Katakis, G. Tsoumakas, and I. Vlahavas, "Tracking recurring contexts using ensemble classifiers: An application to email filtering," *Knowl. Inf. Syst.*, vol. 22, pp. 371–391, 2010, doi: 10.1007/s10115-009-0191-3.

[39] S. Ramírez-Gallego *et al.*, "Survey on data preprocessing for data stream mining: Current status and future directions," *Neurocomputing*, vol. 239, pp. 39–57, 2017, doi: 10.1016/j.neucom.2017.01.078.

[40] A. Bifet, G. Holmes, B. Pfahringer, and R. Kirkby, "MOA: Massive online analysis," *J. Mach. Learn. Res.*, vol. 11, pp. 1601–1604, 2010.

[41] D. Dua and C. Graff, "UCI machine learning repository," Univ. California, Irvine, School of Information and Computer Sciences, 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[42] A. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Secur. Defense Appl.*, 2009, pp. 1–6, doi: 10.1109/CIDSA.2009.5356528.

[43] M. Tavallaee, N. Stakhanova, and A. A. Ghorbani, "Toward credible evaluation of anomaly-based intrusion-detection methods," *IEEE Trans. Syst. Man Cybern. C*, vol. 40, no. 5, pp. 516–524, 2010, doi: 10.1109/TSMCC.2010.2048428.

[44] M. Ring, S. Wunderlich, D. Scheuring, D. Landes, and A. Hotho, "A survey of network-based intrusion detection data sets," *Comput. Secur.*, vol. 86, pp. 147–167, 2019, doi: 10.1016/j.cose.2019.06.005.

[45] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. ICISSP*, 2018, pp. 108–116, doi: 10.5220/0006639801080116.

[46] I. Sharafaldin, A. H. Lashkari, S. Hakak, and A. A. Ghorbani, "Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy," in *Proc. ICISSP*, 2019, pp. 1–9, doi: 10.5220/000736450001009.

[47] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *Proc. Mil. Commun. Inf. Syst. Conf. (MilCIS)*, 2015, pp. 1–6, doi: 10.1109/MilCIS.2015.7348942.

[48] N. Moustafa and J. Slay, "The evaluation of network anomaly detection systems: Statistical analysis of the UNSW-NB15 data set and the comparison with the KDD99 data set," *Inf. Secur. J. Glob. Perspect.*, vol. 25, no. 1–3, pp. 18–31, 2016, doi: 10.1080/19393555.2015.1125974.

[49] M. Habibi Lashkari, G. Draper-Gil, M. Mamun, and A. A. Ghorbani, "Characterization of Tor traffic using time based features," in *Proc. ICISSP*, 2017, pp. 253–262, doi: 10.5220/0006105602530262.
[50] A. Shiravi, H. Shiravi, M. Tavallaee, and A. A. Ghorbani, "Toward developing a systematic approach to generate benchmark datasets for intrusion detection," *Comput. Secur.*, vol. 31, no. 3, pp. 357–374, 2012, doi: 10.1016/j.cose.2011.12.012.
[51] S. Li, W. Meng, and W. Li, "Design of intrusion detection system based on anomaly behavior," *J. Phys. Conf. Ser.*, vol. 1237, p. 032020, 2019, doi: 10.1088/1742-6596/1237/3/032020.
[52] M. A. Ferrag, L. Shu, X. Yang, A. Derhab, and L. Maglaras, "Security and privacy for green IoT-based agriculture: Review, blockchain solutions, and challenges," *IEEE Access*, vol. 8, pp. 32031–32053, 2020, doi: 10.1109/ACCESS.2020.2973178.
[53] Y. Xin *et al.*, "Machine learning and deep learning methods for cybersecurity," *IEEE Access*, vol. 6, pp. 35365–35381, 2018, doi: 10.1109/ACCESS.2018.2836950.
[54] W. Wang, Y. Sheng, J. Wang, X. Zeng, and J. Ye, "HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection," *IEEE Access*, vol. 6, pp. 1792–1806, 2018, doi: 10.1109/ACCESS.2017.2779270.
[55] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, 2018, doi: 10.1109/TETCI.2017.2772792.
[56] R. Vinayakumar, K. Soman, and P. Poornachandran, "Applying convolutional neural network for network intrusion detection," in *Proc. Int. Conf. Adv. Comput. Commun. Informatics (ICACCI)*, 2017, pp. 1222–1228, doi: 10.1109/ICACCI.2017.8126009.
[57] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017, doi: 10.1109/ACCESS.2017.2762418.
[58] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, pp. 41525–41550, 2019, doi: 10.1109/ACCESS.2019.2895334.
[59] Y. Zhang, P. Chen, X. Guo, Z. Lin, and Y. Yu, "Deep learning for network intrusion detection: A survey," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2019, pp. 1–6, doi: 10.1109/INFCOMW.2019.8845074.
[60] R. Vinayakumar, K. P. Soman, and P. Poornachandran, "A deep learning approach for intelligent network intrusion detection system," in *Proc. IEEE Int. Conf. Intell. Secur. Inform. (ISI)*, 2017, pp. 1–6, doi: 10.1109/ISI.2017.8004872.

p-ISSN: **2656-5935** http://journal-isi.org/index.php/isi e-ISSN: **2656-4882**

[61] Z. Wang, "Deep learning-based intrusion detection with adversaries," *IEEE Access*, vol. 6, pp. 38367–38384, 2018, doi: 10.1109/ACCESS.2018.2854609.
[62] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of deep reinforcement learning to intrusion detection for supervised problems," *Expert Syst. Appl.*, vol. 141, p. 112963, 2020, doi: 10.1016/j.eswa.2019.112963.
[63] Y. Liang, K. P. Chow, K. H. Pun, and H. C. Chan, "Deep reinforcement learning for network intrusion detection," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2020, pp. 1–6, doi: 10.1109/ICC40277.2020.9148869.