Detecting Zero-Day Attacks under Concept Drift: An Online Unsupervised Threat Detection System

Article

Accepted version

A. De Paola, S. Drago, P. Ferraro, G. Lo Re

Proceedings of the Italian Conference on Cybersecurity (ITASEC 2024) - CEUR WORKSHOP PROCEEDINGS

Publisher: CEUR-WS

Detecting Zero-Day Attacks under Concept Drift: An Online Unsupervised Threat Detection System

Alessandra De Paola[1,2], Salvatore Drago[3],
Pierluca Ferraro[1,2], and Giuseppe Lo Re[1,2]

[1] Department of Engineering, University of Palermo, Italy
[2] Cybersecurity National Lab, CINI - Consorzio Interuniversitario Nazionale per l'Informatica
[3] IMT School for Advanced Studies Lucca, Italy
alessandra.depaola@unipa.it, salvatore.drago@imtlucca.it
pierluca.ferraro@unipa.it, giuseppe.lore@unipa.it

Abstract

In recent years, there has been significant interest towards mechanisms for detecting cyber-security threats. However, the dynamic nature of modern systems and networks poses significant challenges for threat detection systems exploiting machine learning models, since shifts in data's statistical distribution over time, known as concept drift, can cause severe performance degradation. In this scenario, traditional static systems often need manual retraining by human operators, leaving networks exposed to vulnerabilities in the interim. Moreover, the challenge of detecting zero-day attacks through semi-supervised or unsupervised models remains a critical aspect that has garnered much attention in the literature. This work introduces an unsupervised online threat detection system designed to identify anomalous traffic indicative of zero-day attacks, while explicitly handling concept drift by automating retraining processes only when necessary. An extensive experimental evaluation on the real-world IoT-23 dataset, encompassing network traffic from IoT devices and malicious traffic from malware-infected devices, showcases the system's efficacy, showing superior performance in real-time threat detection compared to traditional static approaches.

1 Introduction

The digital age has seen an exponential rise in cyber-attacks targeting networks, computers, information systems, and IoT devices. Consequently, cybersecurity has become essential as the integration of such technologies into our daily lives continues to evolve, and ensuring their security is paramount. As a result, interest in developing systems for detecting cybersecurity threats has grown steadily. Intrusion Detection Systems (IDS) are a prime example of systems extensively studied, and many works [1, 2] in the literature have demonstrated remarkable performance in distinguishing between benign and malicious traffic, exploiting supervised machine learning algorithms like deep neural networks [3] and decision trees [4, 5]. These algorithms are capable of learning complex separation criteria between different types of traffic and generalizing from the training dataset.

Yet, the rapid evolution and growing complexity of digital systems pose new challenges. The dynamic nature of these environments, compounded by the continuous emergence of new threats, particularly zero-day attacks, necessitates more adaptable solutions. Recent research has thus concentrated on two areas in particular. The first is the development of zero-day detection mechanisms using semi-supervised models that can accurately classify malicious data by detecting differences from the benign data on which they are trained. The second area concerns the phenomenon of concept drift [6, 7], where the statistical properties of input data

may unexpectedly change over time, leading to the obsolescence of previously trained machine learning models and the emergence of new errors and inaccuracies. Such drifts can occur, for example, when the benign traffic pattern of a corporate network shifts due to alterations in network topology, infrastructure changes, the introduction or removal of services, software updates, or changes in employee behavior prompted by new corporate policies. Notably, zero-day attacks can significantly deviate from the statistical distribution of known malicious traffic, thus often evading detection by even the most sophisticated supervised systems. In real-world scenarios, these changes can be sudden, and the same type of statistical input distribution, or *concept*, can be recurrent. Consider, for example, a *click-day* event or the shift from in-person to remote work or education and vice versa, during pandemic periods.

Responding to these challenges, this work introduces a novel online unsupervised anomaly detection system aimed at identifying zero-day attacks within network environments, with a specific focus on addressing concept drifts. The system's architecture is multi-layer and incorporates drift detection mechanisms alongside an unsupervised anomaly detection model. The primary objective is to reduce the necessity for frequent model re-training in response to concept drift, while simultaneously maintaining high detection accuracy. A distinctive feature of the proposed approach is the management of *recurring* concept drifts, a common occurrence in dynamic real-world scenarios. Unlike conventional drift-aware systems, which may discard outdated models, our system retains them for potential future reuse, recognizing the cyclic nature of certain drift patterns. A new model is trained only when all existing models prove to be inadequate for incoming data streams, ensuring efficient use of resources.

The effectiveness of the proposed system was extensively evaluated using the IoT-23 dataset [8], which includes real-world data from IoT devices, allowing for a rigorous evaluation in realistic conditions. The system is also compared to a static approach to demonstrate its effectiveness, highlighting its ability to adapt to novel threats with high accuracy and with reduced re-training frequency, while being fully unsupervised.

The principal contributions of this paper are summarized as follows: (1) introduction of an online unsupervised anomaly detection system specifically designed to handle recurring concept drifts in zero-day attack scenarios; (2) adaptation of the traditional technique of semi-supervised static anomaly detection with AutoEncoder for unsupervised real-time data stream analysis; (3) reduction in the frequency of required re-training for anomaly detection models, without compromising detection accuracy; (4) comprehensive validation of the proposed system using a real-world dataset.

The remainder of the paper is structured as follows. Section 2 reviews related work, while Section 3 details the proposed architecture. Section 4 presents the experimental setup and findings, leading to Section 5, where we draw our conclusions and suggest directions for future research.

2 Related Work

In modern networks, which include a mix of personal and IoT devices, the distribution of input or output data can change unpredictably over time. This phenomenon, known as concept drift, has been identified as a significant challenge in machine learning and cybersecurity [9, 10]. Concept drift can arise from various sources and is broadly categorized into two types: virtual drift, where the distribution of input data changes without affecting the output predictions, and actual drift, which is characterized by stable input data distributions but changing predictions, leading to alterations in the decision boundary. Often, these drifts occur at the same time, resulting in simultaneous changes in both input data distribution and output predictions.

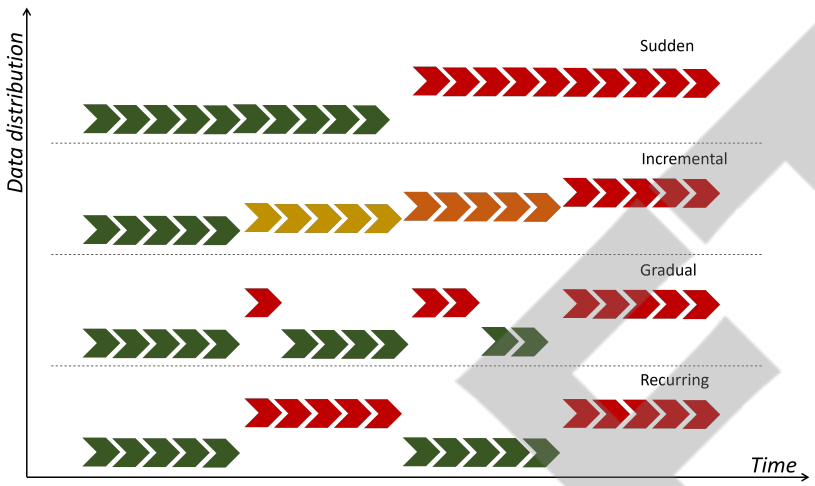

Figure 1: Types of Concept Drift.

Another critical classification of concept drift scenarios is based on the evolution pattern of data distribution over time, typically identified as sudden, incremental, gradual, and recurring drifts, each with distinct characteristics, as shown in Fig. 1.

Despite the growing number of studies, research explicitly addressing both threat detection systems and concept drift remains relatively scarce. The authors of [9] describe the most commonly used workflow in literature for managing concept drift via sliding windows and three distinct phases: real-time evaluation of each new record by the model, detection of concept drift, and adaptation to any detected drift. Concept drift detection is typically based on the model's error rate; a system's performance degradation is attributed to concept drift, triggering an adaptation phase that often includes supervised retraining with data causing the performance decline. However, this approach's reliance on error rates for drift detection requires knowledge of the actual data labels for comparison with predictions, a requirement that is impractical in online contexts due to the unrealistic expectations for real-time labeling by domain experts.

This challenge is common across the literature on intrusion detection methods that are aware of concept drift, as seen in works such as [11]. The proposed system overcomes this limitation by adopting an unsupervised approach for detecting both concept drifts and anomalous traffic.

In the context of unsupervised drift detection, the authors of [12] assess incoming data in batches, using a parameter to limit the anomaly rate in these batches. However, selecting an appropriate value for this parameter is difficult without additional data knowledge in an unsupervised setting. The authors of [13] address this issue by recommending the use of KSWIN [14] for incoming data feature drift detection, allowing for model adjustments prior to the evaluation.

A notable gap in these methodologies is their failure to account for the potential recurrence of concept drifts. Unlike these approaches, our system is designed to detect and manage recurring concept drifts effectively, reducing the frequency of retraining phases. Moreover, many anomaly detection systems discussed in the literature are not well-suited for online applications.

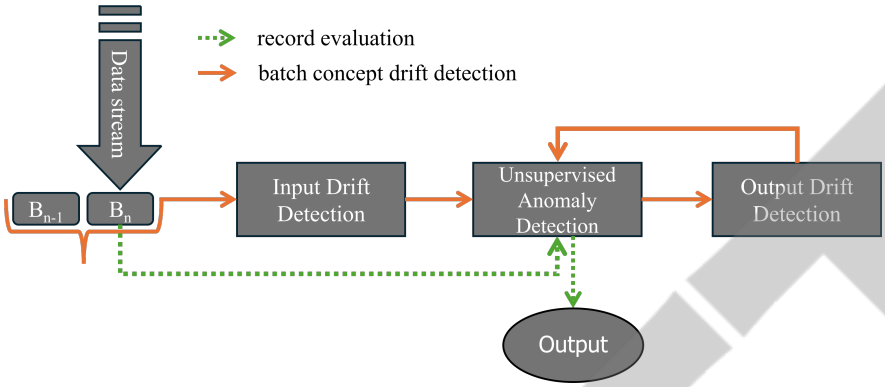

Figure 2: Proposed Architecture.

For detecting zero-day attacks, AutoEncoders are often used in a semi-supervised manner [15] to model benign traffic data's principal characteristics, which constitutes the training set. Additionally, a threshold for distinguishing between benign and malicious traffic is empirically defined during the training phase. This procedure does not fit the operational requirements of the proposed system for online functionality, as detailed in Section 3.1. To address this, we have adapted advanced anomaly detection techniques using AutoEncoders to enhance the real-time analysis of streaming data.

3 Proposed Architecture

This section presents the architecture of the unsupervised threat detection system, as illustrated in Fig. 2. The architecture integrates three core components: the Input Drift Detection (*IDD*) module, which monitors the statistical distribution of incoming data; the Unsupervised Anomaly Detection (*UAD*) module; the Output Drift Detection (*ODD*) module that detect drifts based on the anomaly scores of such models. These components allow the system to handle sudden and recurring concept drift, autonomously identifying shifts in data distribution indicative of new, previously unencountered *concepts* that were not captured during the initial training phase of the model.

To mitigate performance degradation caused by such drifts, the system checks its memory for a pre-existing model capable of addressing the new *concept*, thereby minimizing the need for retraining. In the absence of an applicable model, the system proceeds to train a new one tailored to the data causing the concept drift. Furthermore, the architecture is designed to manage streaming data, addressing the inherent challenges of real-time data processing.

A practical implementation of this architecture could be envisioned in a server setup dedicated to the continuous monitoring of network traffic, aiming to detect and isolate suspicious activities that may signify compromised devices. To this end, the design of the system also factors in memory and temporal constraints typical of real-time threat detection scenarios.

To account for temporal constraints, data processing is performed in a sequential manner. Such a methodology significantly reduces the latency between the acquisition of new data and its subsequent analysis, allowing the system to provide predictions on novel samples not previously used for training. This approach, often referred to as the "test-then-train" approach, is commonly known as Prequential Evaluation [16]. As soon as a new record is available,

the system immediately evaluates it with the anomaly detection model currently in use, as indicated by the green dotted path in Fig. 2. This data is then saved for potential concept drift identification and, if required, for the retraining of the model, as depicted by the orange continuous path in Fig. 2.

To address memory limitations, a fixed-size sliding window technique is used; the adopted approach ensures that only the two most recent batches of data are retained at any point in time, effectively managing the system's memory resources. The sliding window W used by the system has a constant dimension $2w$, containing the two most recent data batches B_{n-1} and B_n, where each batch B_i contains w elements. Additionally, the system is constrained to maintain a finite set of N models within *UAD*, alongside an array of anomaly scores derived from the data on which these models were originally trained.

The first unsupervised anomaly detection model used by the system is trained on the first batch of data, B_0. From that point on, each new record x_i is evaluated and labeled through the current model before being incorporated into the sliding window W. When a new batch of data B_n has been stored in W, the system tries to detect potential drifts through the *IDD* module. If *IDD* detects concept drift between the last two batches of data, then B_n is re-evaluated by each model in *UAD* to obtain the lists of anomaly scores for each model.

Hence, for each model, the system retains anomaly score lists for both the training data and the new data that caused the drift. The *ODD* detection module is then employed to analyze these lists for each model. In scenarios where drift is detected for all models, a new model is trained with B_n and it is set as the currently used model for future evaluations. If the list of models is full, the oldest one is replaced to accommodate the new one. Conversely, if a pre-existing model in *UAD* is found to be drift-resistant by *ODD*, it is selected as the model to use.

Our methodology for incorporating concept drift detection into both the *IDD* and *ODD* modules utilizes the KSWIN algorithm [14], which is based on the Kolmogorov-Smirnov (KS) statistical test principles [17]. KS is a non-parametric test, useful for its independence from underlying data distribution assumptions, although it traditionally applies only to one-dimensional data by evaluating the maximum difference in the distribution functions of two sets of data. The KSWIN method can be implemented within the *ODD* module by comparing one-dimensional distributions of reconstruction errors. Given the multidimensional nature of our data, we adapted KSWIN for *IDD* with a necessary modification, as advised in [13], by applying the algorithm individually to each feature. Whenever drift is detected in any feature, the system considers the need to either initiate training on a new model or reinstate a previously trained model.

3.1 Unsupervised Anomaly-Detection: AutoEncoder

In this work, an AutoEncoder (AE) has been chosen as the base model for unsupervised anomaly detection within the *UAD* module. AutoEncoders are specialized forms of artificial neural networks adept at learning efficient and meaningful encodings of a dataset's feature space. This is achieved through two principal functions: an encoder that transforms the input into a reduced encoded form, and a decoder that attempts to recreate the original input from this encoded representation.

For anomaly detection purposes, AutoEncoders are generally employed in a semi-supervised manner to model and replicate the principal characteristics of benign traffic data, which forms the basis of the training set. The dataset is initially partitioned into training and testing subsets, with the training subset further refined to include only benign records. These records

Algorithm 1: AutoEncoder training.

Input :
B: batch of data used for training;
Output:
$model$: new AutoEncoder model with trained encoder (e_ϕ) and decoder (d_θ);
sc: Standard Scaler of $model$;
α: threshold.

$model \leftarrow$ AE()
ϕ,$\theta \leftarrow$ Random weights initialization
$sc \leftarrow$ StandardScaler()
$sc \leftarrow$ sc.fit(B)
$Bt \leftarrow$ sc.transform(B)
$Train, Validation \leftarrow$ split(Bt)
$model, \phi$,$\theta \leftarrow$ model.fit(Train, Validation)
$r \leftarrow$ model.predict(Bt)
$\alpha \leftarrow \mu(r) + 2\sigma(r)$

Algorithm 2: AutoEncoder testing.

Input :
x_i: i-th record of network traffic;
$model$: selected AE model with encoder(e_ϕ) and decoder(d_θ);
sc: Standard Scaler of selected $model$;
α: threshold.

$xs_i \leftarrow$ sc.transform(x_i)
$r(i) \leftarrow$ model.predict(xs_i)
if $r(i) > \alpha$ **then**
  x_i is anomaly
end
else
  x_i is not anomaly
end

are used to fine-tune the AutoEncoder, enabling it to reconstruct benign traffic data accurately while failing to reconstruct anomalous data, which it has not encountered during training. This discrepancy in its reconstruction capability, particularly evident during the testing phase when malicious records are introduced, signals the presence of anomalies [15].

The reconstruction error is defined as follows:

$$r(i) = \|x_i - d_\theta(e_\phi(x_i))\|. \tag{1}$$

Equation 1 is computed as the absolute difference between the original data point x_i and its reconstruction $d_\theta(e_\phi(x_i))$, where d_θ denotes the decoding function and e_ϕ the encoding function. During the testing phase, this metric is employed as an anomaly score; if the score exceeds a threshold value α, the record is deemed anomalous. The threshold value α is determined based on the model's performance across the entire training dataset, with the objective of effectively distinguishing between benign and malicious traffic.

Unlike the semi-supervised static operation described so far, in the proposed system the AutoEncoder is utilized in a fully unsupervised manner. The online nature of the problem and the specific application domain preclude the assumption that the system can access ground truth data about the current window in time for training. Therefore, it is not feasible to guarantee that the model is exposed exclusively to benign traffic. However, it can be assumed that, in a fixed period, the frequency of anomalies compared to the entire set of observations

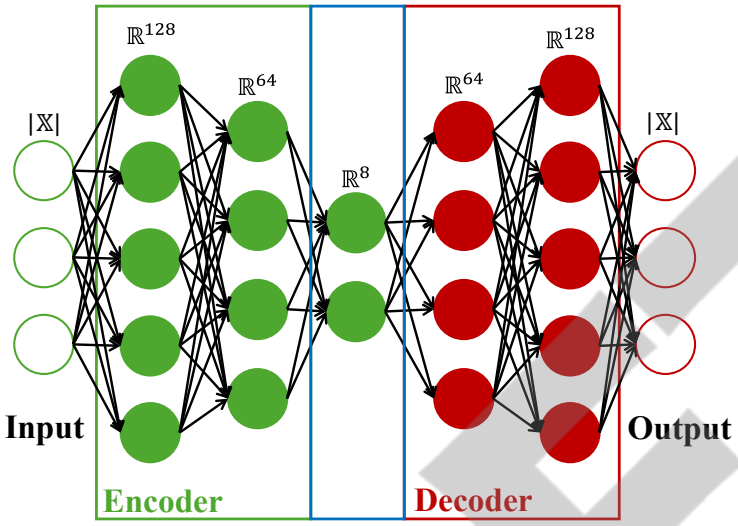

Figure 3: AutoEncoder architecture.

is very low, to such an extent that the impact of these anomalies during the training phase is negligible.

In this work, all the AutoEncoders used share the structure described in Fig. 3. The encoder comprises two hidden layers with dimensions of 128 and 64, respectively, while the decoder is structured with two hidden layers having dimensions of 64 and 128, respectively. The latent space is characterized by 8 dimensions. The Gaussian Error Linear Unit (GELU) serves as the activation function for all layers except for the output layer, which employs a linear function. During the training phase, the Mean Absolute Error (MAE) related to the reconstruction error is used as the loss function. The MAE is defined as follows:

$$MAE = \frac{\sum_{i=0}^{N} r(i)}{N} \tag{2}$$

A detailed description of the training phase is provided in Algorithm 1. Lines 1 and 2 initialize a new AutoEncoder with the previously described structure and random weights for both the encoder and decoder. Line 3 involves the creation of a new Standard Scaler, which is a preprocessing tool that normalizes the data by removing the mean and scaling to unit variance, ensuring that each feature contributes equally to the distance computations. This is crucial for models sensitive to the magnitude of features. Each AutoEncoder in the set of models available to the system is paired with a Standard Scaler, fitted on the batch of data on which the new model is to be trained (line 4), and is then used to transform the training data batch (line 5) as well as each new data record before it is evaluated by the model.

The training data batch is first scaled and then split between the training set (80%) and the validation set (20%), as shown in line 6. After the training phase (line 7), the reconstruction error r, as defined in Eq. 1, is calculated for each data point in the training batch (line 8) and used to determine the threshold value α (line 9).

Unlike in semi-supervised operation, the selection of α cannot be based on the system's performance on the training batch in terms of correctly labeling anomalous traffic, since ground truth data is not available. In this work, assuming that the distribution of the reconstruction

error r for the training data, and thus for benign traffic, can be approximated by a normal distribution, α is set to $\mu(r)+2\sigma(r)$, the mean plus two standard deviations of the reconstruction error distribution. This choice has been deemed appropriate based on experimental evaluations reported in Section 4.

Algorithm 2 describes the evaluation phase for a new record. In line 1, the record x_i is transformed using the specific model's Standard Scaler, then, in line 2, the reconstruction error $r(i)$ is calculated. If $r(i)$ exceeds the threshold value α (line 3), chosen during the training phase for that model, it is labeled as an anomalous traffic record (line 4); otherwise, it is labeled as benign traffic (line 7).

4 Experimental Evaluation

This section evaluates the proposed system's performance utilizing a comprehensive suite of metrics, including accuracy, F1-score, false positive rate (FPR), and false negative rate (FNR), in line with established scientific standards.

The experiments leverage the IoT-2023 dataset collection [8], a repository of network traffic data including heterogeneous IoT devices in both secure and compromised conditions due to various malware threats. Specifically, the experiments are performed on a modified subset of the IoT-2023 collection, integrating data from an expanded range of IoT devices and malware instances. This subset includes both sudden and recurring concept drifts, presenting a challenging environment for the evaluation of a threat detection system. Given the system's online and unsupervised nature, and its initial training on traffic presumed mainly benign, all attacks encountered in the experiments are treated as unknown, similar to zero-day attacks. This feature highlights the system's capability to identify and respond to novel threats without prior knowledge, simulating a real-world scenario where each attack poses a novel challenge.

To address the significant class imbalance present in the dataset, the weighted F1-score was adopted as the metric of choice. Unlike the standard F1-score, which does not account for true negatives and may provide a skewed perspective in the presence of unbalanced classes, the weighted F1-score incorporates the prevalence of each class. This adjustment ensures a more accurate representation by emphasizing the importance of more frequently occurring classes [18].

All experiments utilize the AutoEncoder (AE) architecture described in Sec. 3.1. The AE models, developed using Keras and TensorFlow, are trained through backpropagation, employing the Adam optimizer with a learning rate set to 0.001. To enhance training efficiency and prevent overfitting, an early stopping mechanism was implemented. This mechanism, activated after the tenth epoch, monitors the Mean Absolute Error (MAE) on the validation set, with a patience parameter of 5 epochs, terminating training if no improvement is observed. The training is further bounded by a maximum duration of 100 epochs or 300 seconds.

Experiments were repeated by varying several hyperparameters of the AE architecture, including the size w of the data window (from a minimum of 150 to a maximum of 500), the batch size during training (ranging between 16 and the largest power of two less than w), and the maximum number of stored models N in the UAD module (ranging from 1 to 15). Moreover, the experiments were also conducted with two different methodologies in addition to the proposed one: a conventional static approach, which involves a single initial training phase with no drift detection technique (referred to as *Static*), and a dynamic approach where the model is retrained upon each detected concept drift without explicitly addressing recurring concept drift (referred to as *Detect Drift and Retrain*, or *DDR*).

Results, illustrated in Figure 4, highlight the superior performance of the proposed system under optimal parameter configurations for each evaluated method. Notably, the *static*

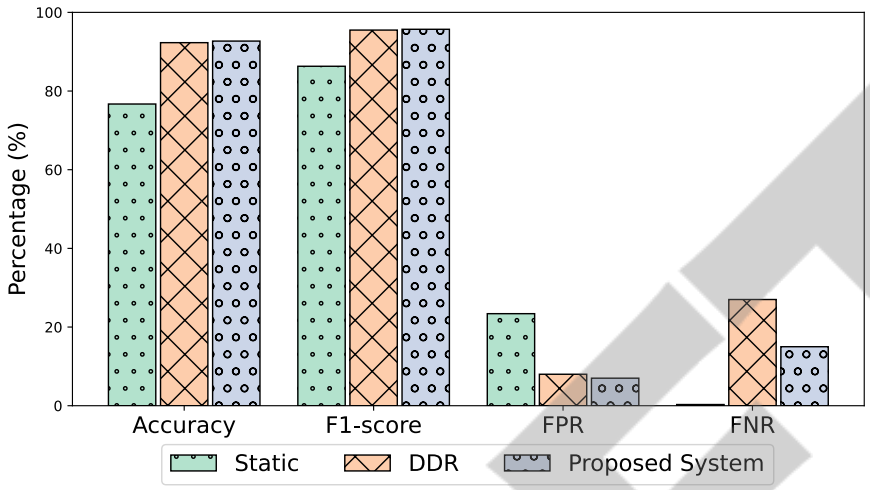

Figure 4: Accuracy, F1-score, FPR and FNR of the static approach, DDR and proposed system.

approach exhibits a high False Positive Rate (FPR), misclassifying 23.4% of benign traffic as malicious. This high value arises despite selecting a threshold α aiming for a 5% FPR, based on the assumption of a normal distribution of benign traffic, as discussed in Sec. 3.1. Conversely, both the DDR and the proposed system closely align with this FPR target, achieving rates of 8% and 7%, respectively, showing their efficacy in adapting to dynamic traffic patterns.

These findings highlight the dynamic nature of benign traffic, which evolves over time, leading to concept drifts. When traffic patterns gradually change, the performance of the static system degrades, because its model and α threshold no longer reflect the current input data distribution. This results in lower accuracy and F1-score (76.7% and 86.3%, respectively), compared to the other approaches. Indeed, systems equipped to adapt to these shifts in distribution by detecting and adjusting to concept drift maintain their performance efficacy. The DDR approach and the proposed system obtain accuracy rates of 92.3% and 92.7%, and F1-scores of 95.5% and 95.7%, respectively, demonstrating their superior adaptability over the static model. Both methods exhibit an increase in the FNR, attributable to the misidentification of concept drift by the IDD module amidst data noise, leading to the detection of a non-existent concept drift and thus unnecessary model retraining and threshold recalibration.

The proposed system, however, mitigates this issue, achieving a lower FNR of 15% compared to DDR's 27%. By recognizing the current model as appropriate for the "false drift" detected by the *IDD* module, the proposed system avoids unnecessary retraining and threshold adjustments. This capability results in slightly higher accuracy results (92.7%) compared to *DDR* (92.3%) while reusing an existing model 30% of the time, thus optimizing computational resource usage. Minimizing evaluation delay is crucial for online threat detection systems, especially in environments where data arrival patterns are unpredictable, as detailed in Sec. 3.

Table 1 illustrates the execution times of the three approaches, relative to the static system that is considered as a baseline (1.0x execution time). While the static system exhibits the fastest execution time due to its lack of retraining, the trade-off in accuracy (92.3% for the proposed system versus 76.7% for the static system) justifies the slight increase in execution time (14% on average). The proposed system represents an optimal compromise, as it significantly outperforms the *static* approach and slightly exceeds the performance of the DDR system in

Table 1: Execution times of the three approaches.

Drift Detection Strategy	Reused Models	Time
Static	-	1.00x
Always retrain (DDR)	0%	1.19x
Proposed system	30%	1.14x

terms of accuracy and F1-score, while exhibiting reduced execution times compared to the *DDR* approach that continually re-trains without addressing recurring concept drifts.

5 Conclusions and Future work

This work explored the challenges of detecting and managing concept drift in the realm of threat detection for network traffic streams. A novel system was introduced, combining unsupervised anomaly detection with concept drift detection techniques for increasing its resilience against zero-day attacks. The system applies AutoEncoder anomaly detection algorithms, typically used in static, semi-supervised contexts, to an online unsupervised setting, enhancing its practical usability in streaming data scenarios. A key contribution of the proposed approach is the explicit management of recurring concept drifts by maintaining a repository of previous models, significantly reducing the need for frequent re-training.

The validity of the proposed methodology was rigorously tested through comprehensive evaluations on a real-world dataset. Since the system is trained primarily on benign traffic, it treats all detected attacks as novel, similar to zero-day threats, reflecting real-world conditions where many attacks are unforeseen. The empirical results highlight the system's capability to accurately identify malicious traffic, even in the face of concept drift and unknown attacks, consistently achieving high accuracy and F1 scores. These outcomes not only outperform those of static approaches but also highlight the critical importance of effectively managing concept drift to prevent abrupt performance deterioration. Furthermore, the proposed system's performance exceeds that of a method that always retrain its models in both accuracy evaluations and execution speed, highlighting the often-overlooked significance of addressing recurring concept drift.

For future directions, the system could be enhanced by incorporating a concept drift detection module designed to operate directly in the multidimensional space of input features. This improvement aims to improve the system's resilience to noise and increase efficiency by minimizing false detections of concept drift.

References

[1] Hung-Jen Liao, Chun-Hung Richard Lin, Ying-Chih Lin, and Kuang-Yuan Tung. Intrusion detection system: A comprehensive review. *Journal of Network and Computer Applications*, 36(1):16–24, 2013.

[2] Ravi Vinayakumar, Mamoun Alazab, KP Soman, Prabaharan Poornachandran, Ameer Al-Nemrat, and Sitalakshmi Venkatraman. Deep learning approach for intelligent intrusion detection system. *Ieee Access*, 7:41525–41550, 2019.

[3] Ahmad Javaid, Quamar Niyaz, Weiqing Sun, and Mansoor Alam. A deep learning approach for network intrusion detection system. In *Proceedings of the 9th EAI International Conference on*

Bio-inspired Information and Communications Technologies (formerly BIONETICS), pages 21–26, 2016.

[4] Nabila Farnaaz and MA Jabbar. Random forest modeling for network intrusion detection system. *Procedia Computer Science*, 89:213–217, 2016.

[5] Vincenzo Agate, Felice Maria D'Anna, Alessandra De Paola, Pierluca Ferraro, Giuseppe Lo Re, and Marco Morana. A behavior-based intrusion detection system using ensemble learning techniques. In *ITASEC*, 2022.

[6] Firas Bayram, Bestoun S Ahmed, and Andreas Kassler. From concept drift to model degradation: An overview on performance-aware drift detectors. *Knowledge-Based Systems*, 245:108632, 2022.

[7] Vincenzo Agate, Salvatore Drago, Pierluca Ferraro, and Giuseppe Lo Re. Anomaly detection for reoccurring concept drift in smart environments. In *18th International Conference on Mobility, Sensing and Networking (MSN)*, pages 113–120. IEEE, 2022.

[8] Sebastian Garcia, Agustin Parmisano, and Maria Jose Erquiaga. IoT-23: A labeled dataset with malicious and benign IoT network traffic (1.0.0) [Data set]. Zenodo.

[9] João Gama, Indrė Žliobaitė, Albert Bifet, Mykola Pechenizkiy, and Abdelhamid Bouchachia. A survey on concept drift adaptation. *ACM computing surveys (CSUR)*, 46(4):1–37, 2014.

[10] Jie Lu, Anjin Liu, Fan Dong, Feng Gu, Joao Gama, and Guangquan Zhang. Learning under concept drift: A review. *IEEE Transactions on Knowledge and Data Engineering*, 31(12):2346–2363, 2018.

[11] Deepa Mulimani, Shashikumar G Totad, Prakashgoud Patil, and Shivananda V Seeri. Adaptive ensemble learning with concept drift detection for intrusion detection. In *Data Engineering and Intelligent Computing: Proceedings of ICICC 2020*, pages 331–339. Springer, 2021.

[12] Zhiguo Ding and Minrui Fei. An anomaly detection approach based on isolation forest algorithm for streaming data using sliding window. *IFAC Proceedings Volumes*, 46(20):12–17, 2013.

[13] Maurras Ulbricht Togbe, Yousra Chabchoub, Aliou Boly, Mariam Barry, Raja Chiky, and Maroua Bahri. Anomalies detection using isolation in concept-drifting data streams. *Computers*, 10(1):13, 2021.

[14] Christoph Raab, Moritz Heusinger, and Frank-Michael Schleif. Reactive soft prototype computing for concept drift streams. *Neurocomputing*, 416:340–351, 2020.

[15] Jinwon An and Sungzoon Cho. Variational autoencoder based anomaly detection using reconstruction probability. *Special lecture on IE*, 2(1):1–18, 2015.

[16] Joao Vinagre, Alípio Mário Jorge, Conceição Rocha, and Joao Gama. Statistically robust evaluation of stream-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 33(7):2971–2982, 2019.

[17] Frank J Massey Jr. The Kolmogorov-Smirnov test for goodness of fit. *Journal of the American statistical Association*, 46(253):68–78, 1951.

[18] David MW Powers. Evaluation: From precision, recall and f-factor to roc, informedness, markedness & correlation. *Journal of Machine Learning Technologies*, 2(1):37–63, 2011.