Managing Concept Drift in Online Intrusion Detection Systems with Active Learning

Francesco Camarda[1,†], 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*

Abstract

Machine learning-based Intrusion Detection Systems (IDS) are widely used to identify and mitigate threats by analyzing network traffic for malicious activity. However, most existing IDS solutions assume a closed environment with stable statistical properties. This overlooks challenges posed by open environments and the problem of concept drift, where shifts in network traffic patterns over time can render training data obsolete and degrade the performance of static systems. While online IDS can adapt to these changes, they face the additional challenge of acquiring labeled data in real time, which is often impractical due to time constraints. To address these challenges, this paper proposes an online IDS that employs an incremental supervised Random Forest model combined with a drift-aware approach, designed for open environments with limited labeling. Active learning techniques are used to select the most informative records, minimizing the need for human feedback while retaining enough information to detect drifts. The system adapts incrementally when drift is detected, updating the underlying model as needed. The experimental evaluation, performed on a real-world network dataset, proves the system's effectiveness in open environments and under limited labeling conditions, achieving better performance compared to state-of-the-art methods.

Keywords

Online Intrusion Detection System, Threat Detection, Concept Drift, Active Learning, Incremental Machine Learning

1. Introduction and Related Work

In recent years, cybersecurity has received increasing attention, especially in the development of advanced threat detection mechanisms [1, 2]. Among these, Intrusion Detection Systems (IDS) are one of the most widely researched tools and play an important role in identifying and mitigating potential threats [3, 4]. The integration of machine learning techniques has significantly advanced the development of automated IDS [5], enabling the analysis of network traffic records extracted from traffic logs to detect malicious activity. For instance, [6, 7] employ deep learning to build supervised systems that identify and categorize malicious traffic. In contrast, other works adopt unsupervised approaches to detect anomalies relative to benign traffic [8, 9] or use Decision Tree ensembles [10, 11], which offer lower training costs and faster predictions than deep learning.

Despite these advancements, several critical challenges remain, highlighting the need for further research to ensure the robustness and practical deployment of machine learning-based IDS. A significant limitation of these systems is their static nature. Most research on machine learning-based IDS assumes a *closed environment*, where the statistical properties of the data-generating process remain stable over time. However, in real-world applications, this assumption often proves unrealistic. Once deployed,

Joint National Conference on Cybersecurity (ITASEC & SERICS 2025), February 03-8, 2025, Bologna, IT

[*] Corresponding author.

[†] These authors contributed equally.

francesco.camarda03@community.unipa.it (F. Camarda); alessandra.depaola@unipa.it (A. De Paola); salvatore.drago@imtlucca.it (S. Drago); pierluca.ferraro@unipa.it (P. Ferraro); giuseppe.lore@unipa.it (G. Lo Re)
0000-0002-7340-1847 (A. De Paola); 0009-0009-0367-0484 (S. Drago); 0000-0003-1574-1111 (P. Ferraro); 0000-0002-8217-2230 (G. Lo Re)

network traffic may not maintain the same statistical distribution as the training data, highlighting the challenges of operating in an *open environment* [12].

Consider, for example, an IDS designed to monitor the network traffic of a university or private company. Initially, the system might be trained on a dataset of labeled benign and malicious traffic, compiled by domain experts during typical daily activities. However, the COVID-19 pandemic disrupted traffic patterns as organizations shifted from in-person to online activities (e.g., virtual classes, exams, and meetings) and later reverted to hybrid or in-person models, introducing new platforms and services not included in the training data.

This scenario illustrates the phenomenon of *concept drift*, specifically *recurring* drift, where the data generation process becomes non-stationary [13, 14]. Such shifts in network traffic render previous training data obsolete, introduce errors, and degrade performance in static systems [15]. Addressing these changes often requires manual retraining, leaving networks vulnerable during this period.

A highly effective approach to deal with open environments is *online learning* [16], according to which data streams are processed in real time, while specific algorithms detect anomalies and the occurrence of concept drift [17]. The authors of [18] propose a continuous learning adaptation of deep neural networks (DNNs), dynamically adjusting the network size using a hedge weighting mechanism. Similarly, the authors of [19] introduce an online adaptation of the Local Outlier Factor (LOF) anomaly detection model to handle recurring concept drift and minimize retraining phases.

Current adaptation techniques [20] fall into two categories: *detect and retrain*, which discards the old model and retrains on new data, and *detect and update*, which refines the model incrementally [21]. This distinction reflects the *stability-plasticity dilemma* [22, 23], which refers to the challenge of balancing knowledge retention and learning new concepts. These two operations are inherently opposed, and current state-of-the-art adaptation techniques tend to be overly biased toward one approach over the other.

To address this dilemma, the authors of [24] propose an incremental adaptation of a Decision Tree ensemble-based machine learning model, where new members are added or existing ones are replaced within the ensemble. Building on this idea, the authors of [25] present a system to handle concept drift by incorporating a pruning strategy and weighted voting of individual trees based on prediction error, achieving a trade-off between stability and plasticity.

However, these works overlook the *cost of labeling* problem. Both drift detection and adaptation phases assume that ground truth labels become available after a certain interval. During detection, drift is identified by comparing predicted labels with ground truth labels to detect performance degradation. In the adaptation phase, supervised techniques rely on ground truth data to retrain or update the model when drift is detected. However, this additional labeling phase depends on real-time feedback from human experts, which is often impractical due to constraints of time and volume, particularly in fields like online learning for Intrusion Detection Systems under concept drift.

To overcome this limitation, the authors of [26, 27] propose a novel approach for online learning under concept drift using unsupervised models and detection steps that do not rely on prediction error. This allows handling recurring drift without labeling costs. However, a major drawback of this approach is the reliance on unsupervised models, which can struggle with multi-class classification and high-dimensional data.

Another strategy is to use online supervised models under the assumption of *limited labeling*. In this context, an online IDS is proposed in [28], using only a subset of the data available in order to reduce the burden on human experts. The subsets of retraining data can be selected either through random sampling or *active learning techniques* [29, 30]. The limitation of this approach lies in the need to retrain after each batch, regardless of whether concept drift is present or not. This results in high computational costs and significant labeling effort, even under the limited labeling assumption.

In summary, several aspects of online learning under concept drift remain under-explored, limiting the broader application of this approach in Intrusion Detection Systems.

To address these limitations, the proposed Intrusion Detection System adopts a drift-aware incremental active learning approach, designed to operate effectively in open environments with limited labeling assumptions. This approach uses active learning techniques to select the most informative records, minimizing the need for human expert feedback while preserving sufficient information to detect drifts over time. The system then incrementally updates the underlying machine learning model as necessary.

The proposed system leverages an incremental adaptation of a supervised ensemble-based machine learning model, achieving an optimal balance between stability and plasticity. It adapts rapidly to concept drift while retaining knowledge from previous iterations, which can be reused when needed, such as in cases of recurring drift. This enhances robustness by reducing unnecessary and noisy adaptation phases.

The experimental evaluation, performed on a real network dataset, proves the system's effectiveness in open environments and under limited labeling assumptions, while also showing the impact of concept drift. The proposed approach is compared with other state-of-the-art methods, highlighting differences in overall performance, drift adaptation, and labeling cost, and analyzing how these factors are influenced by the various components of the proposed architecture.

The principal contributions of this paper are summarized as follows: (1) the introduction of an online supervised ML-based Intrusion Detection System designed to operate in open environments and under limited labeling assumptions; (2) the proposal of an incremental Random Forest model that uses active learning to handle concept drift while minimizing the need for human expert feedback; (3) a comparison of different incremental systems employing different active learning techniques under increasingly restrictive limited labeling assumptions; (4) a comprehensive validation of the proposed system using a real-world network dataset affected by concept drift.

The remainder of the paper is structured as follows. Section 2 describes the proposed architecture. Section 3 outlines the experimental setup and presents the findings. Finally, Section 4 draws conclusions and suggests directions for future research.

2. Proposed Architecture

This section presents the architecture of the proposed online Intrusion Detection System, which consists of three main components: an adaptive incremental Random Forest model (*AIRF* ML-model), an informative active learning module (*IAL*), and a concept drift detector (*CDD*). Additionally, the system uses a retraining window (RW) to store recent data for incremental updates. Figure 1 illustrates these components and their interactions.

These components work together within the architecture, which operates in two distinct phases: the *concept drift detection* phase, on the left side of Figure 1, and the *concept drift adaptation* phase, shown on the right side. The paths highlighted in the figure illustrate how data flows through each component and how they interact with each other. These phases enable the system to detect concept drifts, representing new patterns not captured during the initial training, in line with the *open environment* assumption. The architecture also supports the *limited labeling* assumption by minimizing human labeling effort. This is achieved through active learning techniques that select a small percentage of the most informative data for incremental training during the adaptation phase.

The proposed system can be deployed on a server for continuous monitoring of network traffic, with the goal of detecting and isolating suspicious activity. The initial *AIRF* model is trained offline using the first retraining window (RW) and contains a fixed number of trees in the ensemble (*nt*). After this offline training phase, the online process begins, as summarized in Figure 1 and detailed in Algorithm 1.

First, the system evaluates each new record from the data stream using the *AIRF* model, as indicated by the green continuous path in Figure 1. After the prediction, the record is stored in the fixed-size list called *New batch*. When the *New batch* is full, the concept drift detection phase begins. The *IAL* module plays a crucial role in this phase, by identifying the most informative data for model adaptation and selecting a small percentage of records to be labeled, in line with the limited labeling assumption.

This selection process leverages the model's uncertainty to identify potential concept drift. Indeed, in the case of concept drift, the model should exhibit higher uncertainty regarding data from the new distribution. Leveraging this, the adopted strategy maximizes the usefulness of the selected data for

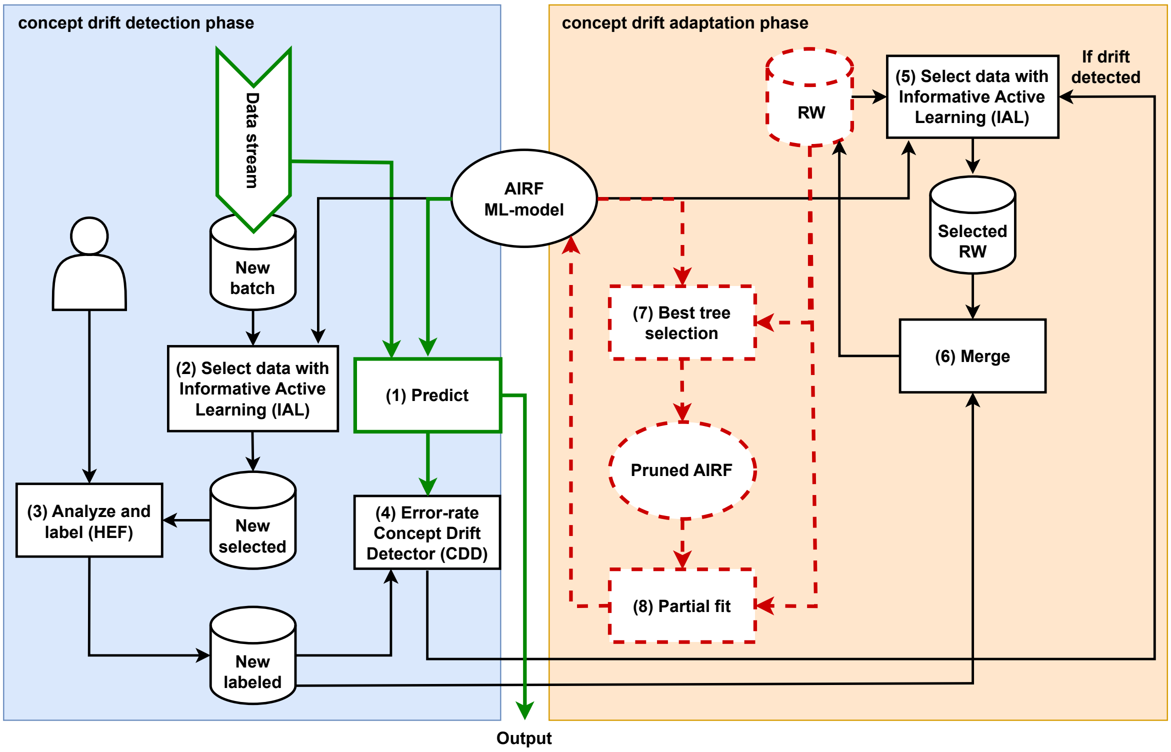
concept drift detection phase
concept drift adaptation phase
Data stream
New batch
AIRF ML-model
(2) Select data with Informative Active Learning (IAL)
(1) Predict
(3) Analyze and label (HEF)
New selected
(4) Error-rate Concept Drift Detector (CDD)
New labeled
Output
RW
(5) Select data with Informative Active Learning (IAL)
If drift detected
Selected RW
(7) Best tree selection
(6) Merge
Pruned AIRF
(8) Partial fit

Figure 1: Workflow of the proposed architecture for the online Intrusion Detection System.

adapting to the drift while minimizing labeling costs, compared to a random sampling strategy, as shown in the experimental section.

In this work, an uncertainty-based informative active learning method is employed to enable rapid adaptation to concept drift. Specifically, the *AIRF* model calculates an uncertainty score for each record as $1 - \max(proba)$, where *proba* is the vector of probability scores assigned to the classes by the model for that record. The records are then sorted in descending order by their uncertainty scores and selected for analysis and labeling by the human expert, forming the *New selected* batch. After receiving human expert feedback (*HEF*), the labeled data (*New labeled*) are processed by the *CDD* module to detect concept drift. The *CDD* is an error-rate concept drift detector that compares the ground truth labels of the *New labeled* batch with the system's predictions. If the system's accuracy falls below a threshold α, concept drift is detected, triggering the adaptation phase.

When this occurs, the *IAL* module is used again to select informative records from the last retraining window (*RW*), equal in number to those in the *New labeled* batch, to complement the newly labeled data. The selected records and the *New labeled* batch are then merged to form a new *RW*. Consequently, the new *RW* contains the most informative data from the previous window and the most informative data associated with the detected concept drift. The proposed incremental training process, shown by the red dotted path in Figure 1, involves evaluating the performance of each tree in the ensemble on the *RW* in terms of accuracy. The best-performing trees are retained, while the others are removed, creating a pruned version of the model (*pAIRF*). A new training phase is then performed on the *pAIRF* using the *RW* window, increasing the number of trees to the expected value *nt*. Finally, the *New batch* list is cleared, and the process restarts.

In the proposed incremental adaptation method, the model remains unchanged during stationary phases when no drift is detected, and it rapidly adapts when drift occurs. After evaluating the individual performance of the ensemble trees on the *RW*, the top-performing half-minus-one trees are selected (*nbt*). The pruned Random Forest is then updated using warm-start techniques to add the missing trees to the ensemble while keeping the existing trees unchanged. The previously described data retention strategy, combined with this incremental approach, allows the model to quickly adapt to concept drift.

Algorithm 1: Proposed system workflow - Online phase.

```
Input  : DS: data stream of network traffic records; bs: size of New_batch list;
         AIRF: proposed adaptive incremental random forest model;
         nt: number of trees in the AIRF; nbt: number of best selected trees ;
         IAL: informative active learning module; %lab: percentage of records to label;
         CDD: concept drift detector; $\alpha$: drift detection threshold; HEF: human expert feedback;
         RW: retraining window (initially contains the data of the first offline training phase).
Output :
         $\hat{Y}$ : the list of system predictions.
New_batch ← [ ]
for $x_i \in DS$ do
    $\hat{y}_i$ ← AIRF.predict($x_i$)                              ▷ prediction performed on new record
    $\hat{Y}$.append($\hat{y}_i$)
    New_batch.append($x_i$)                                        ▷ save the record just predicted
    if len(New_batch)==bs then
        New_selected ← IAL.select(AIRF, New_batch, %lab)           ▷ select the most informative records
        nbr ← len(New_selected)
        New_labeled ← HEF.query(New_selected)                      ▷ labeling by the human expert
        if CDD.detect($\hat{Y}$[New_labeled], New_labeled, $\alpha$) then
            selected_RW ← IAL.select(AIRF, RW, nbr)
            RW ← [New_labeled; selected_RW]                        ▷ update RW
            pAIRF ← select_best_trees(AIRF, RW, nbt)               ▷ create a pruned version of AIRF
            AIRF ← pAIRF.partial_fit(RW, nt)                       ▷ incremental training
        end
        New_batch ← [ ]
    end
end
```

Trees trained on the new data distribution perform a majority vote in the ensemble, instead of following a *detect and update* strategy, which typically involves the slow process of incrementally adapting a deep learning model or an ensemble-based model by adding or replacing a single tree. However, it is beneficial to retain some of the acquired historical information to limit performance degradation caused by unnecessary adaptations with noisy data and to enable faster recovery from recurring concept drift. Compared to a *detect and retrain* strategy, the composition of the *RW* window, combined with incremental training using bootstrap techniques [31] and warm starts, helps to reduce the number of "harmful" trees added during a single incremental training phase. This approach mitigates the impact of sporadic noise, which differs from concept drift due to its limited and non-persistent nature.

In addition to addressing sporadic noise, the strategy of retaining trees and historical data also proves beneficial for managing recurring concept drift effectively. Indeed, in cases of alternating concepts, some trees trained on previous iterations of the recurring concept remain in the ensemble; the best-performing trees are thus preserved, accelerating the system's adaptation to recurring patterns. Finally, the proposed *AIRF* uses a limited number of trees (*nt*) in the ensemble and a fixed-size batch of data, respecting memory constraints and operating effectively under the limited labeling assumption.

3. Experimental Evaluation

This section evaluates the proposed system's ability to handle concept drift effectively, accurately detect malicious activity, and minimize labeling costs. The experiments compare the proposed approach with several other methods, highlighting its advantages in terms of accuracy, adaptability, and efficiency. The performance of the compared systems is evaluated using a comprehensive set of metrics including accuracy, F1-score, true positive rate (TPR), and true negative rate (TNR). These metrics are chosen in accordance with established scientific standards [32] and reflect best practices for evaluating ML-based threat detection systems.

All experiments were conducted on the *KDD CUP'99* dataset, which contains simulated traffic and intrusions from a military network environment and includes various types of benign and malicious traffic. Although some studies [33, 34] have highlighted flaws in this dataset and it may appear outdated compared to more recent network datasets used for validating many static IDSs, it is still considered one of the few real-world network datasets that exhibit sudden and recurring concept drift [13]. These characteristics make it an excellent candidate for testing the performance of ML-based threat detection systems and evaluating online learning strategies under concept drift.

The pre-processing phase involved transforming categorical attributes into numerical ones using one-hot encoding and binarizing the ground truth labels. No additional operations, such as shuffling, PCA, or feature standardization, were applied. These operations are incompatible with the *open environment* assumption and the online nature of the system, as they require prior knowledge of the entire dataset before the offline training phase. In the experiments presented below, the *AIRF* model is trained offline using only the first batch of data (RW_0). After this initial training, the system begins the online process, as described in Algorithm 1 and Section 2. This approach reflects realistic conditions where the system must adapt to new data over time without access to future information during the initial training phase. In contrast to the classic experimental phase of static systems, in which the dataset is partitioned into train and test, in this case there is an initial offline train phase. Subsequently, all data evaluated during the online phase can be considered as test data until the next possible adaptation phase, which involves incremental training of the model with a new retraining window (RW). This approach, often referred to as the "test-then-train" approach, is commonly known as Prequential Evaluation [35].

In addition to the *Proposed* system, experiments were conducted on several other systems that share the same *AIRF* incremental model described in Section 2, but differ in their online strategies:

- *Static*: a non-online system trained once, offline, under the closed environment assumption. This system serves as a baseline for understanding the impact of concept drift.
- *Incremental*: a classical incremental system that retrains the model after each batch, regardless of whether concept drift is present. This approach assumes that the ground truth labels for all the data in the *New batch* and the *RW* lists are available.
- *RSIncremental*: an incremental system similar to *Incremental*, but operating under the limited labeling assumption. It uses a random sampling strategy to select a percentage of records from the *New batch* and the *RW* lists for retraining.
- *IALIncremental*: similar to *RSIncremental*, but employs an informative active learning technique instead of random sampling. This technique selects the most informative records for labeling.
- *RALIncremental*: also similar to *RSIncremental*, but uses a representative active learning technique that selects records closest to the centroids of clusters identified by the K-Means algorithm.

Specifically, *IALIncremental* uses the same active learning method as the proposed system. The key difference is that *IALIncremental* activates the incremental training phase after evaluating each batch, whereas the proposed system activates this phase only when concept drift is detected. To mitigate the influence of randomness, experiments were repeated 1000 times with different random seeds. The results presented are the averages of these tests, ensuring robustness against variations caused by random selection.

All experiments were conducted using the same set of hyperparameters across all compared systems, chosen based on preliminary evaluations to ensure optimal performance and a fair comparison. The *New batch* list size (*bs*) was set to 10000; this value provides optimal performance for the *Static* system during stationary periods and balances the need for timely adaptation in the online systems when concept drift occurs. The number of trees in the *AIRF* model (*nt*) was set to 10, as higher values did not yield further performance improvements. The drift detection threshold (α) was set to 95% accuracy to avoid unnecessary retraining during stationary periods while ensuring the detection of real concept drifts, even though this setting makes the systems more sensitive to noise.

Table 1 presents the best performance for each compared system, showing the relevant metrics along with the percentage of data labeled by the expert (*% labeling*). The standard deviation of the metrics observed in these experiments is negligible and has therefore been omitted for clarity.

Table 1

Comparison of the best results of different systems in terms of: percentage of data labeled by the expert (% labeling), Accuracy, F1-score, True Positive Rate (TPR), and True Negative Rate (TNR).

System	% labeling	Accuracy	F1-score	TPR	TNR
Static	-	80.78%	86.63%	99.98%	76.42%
Incremental	100%	98.48%	99.07%	96.18%	99.00%
RSIncremental	10%	98.35%	98.99%	96.20%	98.84%
IALIncremental	0.5%	96.75%	98.01%	91.79%	97.88%
	10%	96.94%	98.10%	96.98%	96.93%
RALIncremental	10%	98.86%	99.29%	99.22%	98.77%
Proposed	0.5%	98.60%	99.14%	98.78%	98.56%

In particular, the *Static* system shows the worst performance metrics. Although it achieves the highest TPR (99.98%), its overall accuracy is only 80.78%, and its F1-score is 86.63%, with a TNR of 76.42%. These results indicate a significant degradation in performance due to concept drift on benign traffic.

The *Incremental* system proves effective in handling the open environment assumption, achieving very high performance for all metrics presented in Table 1. These results demonstrate that concept drift can be effectively managed using an incremental adaptive approach, such as the one described in Section 2. However, this system suffers from the high cost of labeling, as it requires 100% of the data to be labeled for each incremental training step. This assumption is impractical, as it places an unsustainable burden on human experts.

To consider a more realistic case, the *RSIncremental* system was evaluated under the limited labeling assumption; only a small percentage of the batch is used for incremental training, significantly reducing the labeling cost and maintaining its performance slightly below *Incremental*, with an accuracy of 98.35% and F1-score of 98.99%. The results also show that the *RALIncremental* system, using the same labeling percentage (10%) as *RSIncremental*, achieves better performance compared to random selection, with an accuracy of 98.86% and an F1-score of 99.29%. In contrast, the *IALIncremental* system achieves good results, though slightly lower than *RSIncremental*. Interestingly, *IALIncremental* performs similarly when using either 10% or as little as 0.5% of labeled records. Finally, the *Proposed* system achieves excellent performance, with an accuracy of 98.60% and an F1-score of 99.14%. These results are better than those of the *Incremental* system and only slightly lower than the *RALIncremental* system. However, the *Proposed* system requires only 0.5% of labeled data instead of 10%, and the computational cost of the informative active learning (IAL) method is significantly lower than that of the representative active learning (RAL) method. This ensures a much faster incremental training phase. During the adaptation phase, incoming records are immediately evaluated with the old model. If concept drift is detected, a temporary degradation in performance occurs until the old model is replaced with the adapted one.

Figure 2 shows the performance, in terms of F1-score achieved with a certain percentage of labeled records, of the four compared systems that work under limited labeling assumption. For both *RSIncremental* and *RALIncremental*, the overall F1-score correlates with the percentage of labeled data used for incremental training.

Notably, *RALIncremental* performs worse than *RSIncremental* at lower labeling percentages (0.5%, 1%, 2%, and 4%). However, their performance becomes comparable at 6% and 8%, and *RALIncremental* outperforms *RSIncremental* at 10%. This can be explained by the fact that *RALIncremental* selects records near the cluster centroids. During stationary phases, this strategy effectively mitigates the impact of noisy records by selecting data similar to the previously seen distribution. In these cases, the decision boundary does not need to change significantly, and the system benefits from refining it with new, useful information. However, during concept drift, this approach struggles to select records that accurately represent the new concept, unless the labeling percentage is increased beyond a certain threshold.

IALIncremental achieves excellent performance with 0.5% and 1% of labeled data, but its performance

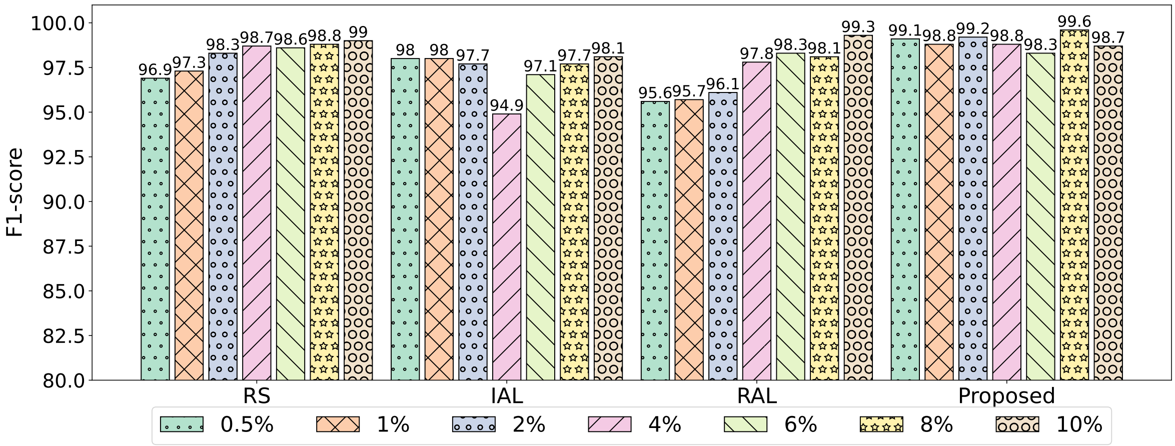

Figure 2: F1-score of the *RSIncremental* (RS), *IALIncremental* (IAL), *RALIncremental* (RAL) and proposed system (*Proposed*) with different percentage of labeling.

decreases, reaching a minimum at 4%, before increasing again at 10%. This behavior occurs because the informative active learning mechanism prioritizes records where the model shows high uncertainty. During concept drift, these uncertain records are beneficial for incremental retraining. However, during stationary phases, the same strategy tends to select noisy records. When the percentage of labeled data is small, the noisy records are still few enough to allow effective adaptation during drift. As the labeling percentage increases, the presence of noisy data becomes more pronounced, causing performance degradation. Beyond a certain threshold, the mechanism also selects records with lower uncertainty, mitigating the negative impact of noise and improving performance. Due to this trade-off, *IALIncremental* achieves its best performance at the smallest (0.5%) and largest (10%) percentages of labeled records.

This also explains the remarkable performance of the *Proposed* system. By activating the incremental training phase only when the *CDD* detects concept drift, the system achieves rapid adaptation using a small number of selected records. This approach avoids unnecessary retraining with noisy data during stationary periods. As a result, the *Proposed* system maintains a more stable performance trend across different labeling percentages, as shown in Figure 2.

Finally, Figure 3 shows the performance trend of the *Static*, *Incremental*, *IALIncremental*, and *Proposed* systems over time, illustrating the average F1-score for each batch. The F1-score for the *Static* system alternates between abrupt drops and periods of high performance during batch changes. These fluctuations demonstrate the severe limitations of the static system under the open environment assumption, where sudden and recurring concept drifts degrade its overall performance, as previously discussed. The *Incremental* system trend shows some negative peaks occurring in the same batches as the *Static* system, confirming the presence of concept drift. However, these drops are less abrupt and less severe compared to those in the *Static* system. Additionally, the *Incremental* system's performance recovers quickly after detecting drift, highlighting the rapid adaptation capabilities of the proposed incremental method, especially under recurring drift conditions. In accordance with the previous considerations, the F1-score of *IALIncremental* increases as quickly as that of the *Incremental* system after concept drift phases. However, during stationary periods, *IALIncremental* exhibits negative performance peaks due to the selection of noisy data, an issue not observed in the *Incremental* system. Finally, compared to *IALIncremental*, the *Proposed* system avoids the negative performance peaks caused by noisy retraining (e.g., batches 3 and 37). When drift is detected, as indicated by the red bands, the system quickly and consistently recovers to optimal performance.

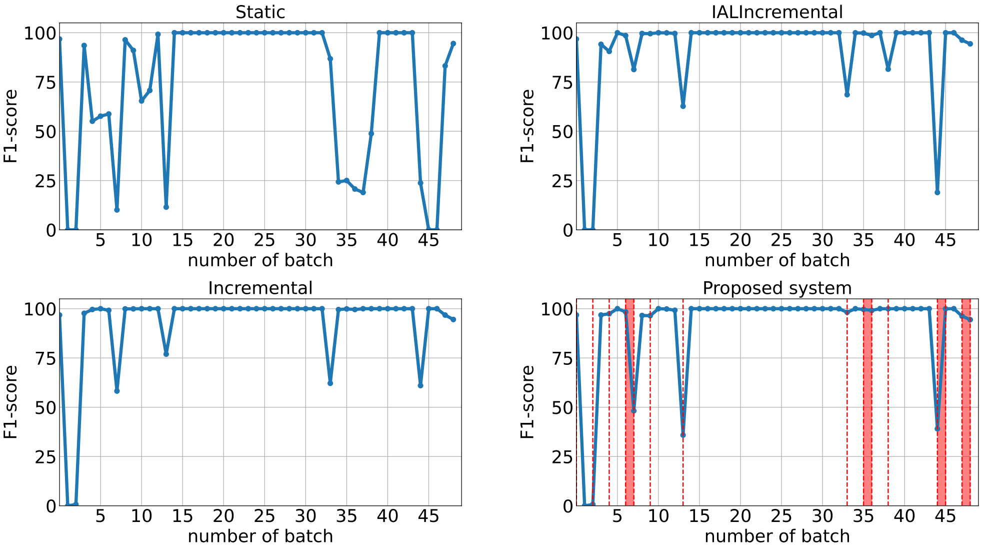

Figure 3: F1-score over number of batch for the Static, Incremental, IALIncremental (0.5%) and Proposed system (0.5%). Vertical red lines indicate that the proposed system has detected concept drift.

4. Conclusions and Future work

This work explored the challenges of detecting malicious activity by analyzing network traffic streams using an online machine-learning-based Intrusion Detection System (IDS) designed for open environments and limited labeling conditions. The primary challenge is detecting concept drift and adapting the model to new data distributions while minimizing the labeling cost. To address these challenges, the proposed system autonomously detects concept drift and activates an adaptation phase using an incremental Random Forest model and an informative active learning technique. This approach ensures optimal adaptation while minimizing the need for human expert feedback. The effectiveness of the proposed methodology was rigorously evaluated using a real-world network dataset with concept drift, under increasingly restrictive limited labeling conditions. The experimental results highlight the robustness of the proposed system, which maintains high and stable performance and prove the system's ability to detect concept drift and accurately identify malicious traffic. The proposed system consistently achieves high accuracy (98.60%) and an F1-score of 99.14%, while requiring only 0.5% of labeled data per batch, outperforming other state-of-the-art techniques. Such results prove the effectiveness of combining concept drift detection, informative active learning, and incremental learning. For future work, the system could be improved by incorporating a more sophisticated unsupervised concept drift detection module that operates directly in the multidimensional space of input features. This enhancement would further reduce the need for human expert feedback and the cost of labeling by triggering active learning only when concept drift is detected.

Acknowledgments

This work was partially supported by the AMELIS project, within the project FAIR (PE0000013), and by the ADELE project, within the project SERICS (PE00000014), both under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] I. H. Sarker, A. Kayes, S. Badsha, H. Alqahtani, P. Watters, A. Ng, Cybersecurity data science: an overview from machine learning perspective, Journal of Big data 7 (2020) 1–29.

[2] V. Agate, A. De Paola, G. Lo Re, A. Virga, Reliable reputation-based event detection in v2v networks, in: International Conference on Advanced Research in Technologies, Information, Innovation and Sustainability, Springer, 2023, pp. 267–281.

[3] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, K.-Y. Tung, Intrusion detection system: A comprehensive review, Journal of Network and Computer Applications 36 (2013) 16–24.

[4] O. H. Abdulganiyu, T. Ait Tchakoucht, Y. K. Saheed, A systematic literature review for network intrusion detection system (ids), International journal of information security 22 (2023) 1125–1162.

[5] V. Agate, F. Concone, A. De Paola, P. Ferraro, S. Gaglio, G. Lo Re, M. Morana, Adaptive ensemble learning for intrusion detection systems, in: CEUR WORKSHOP PROCEEDINGS, volume 3762, CEUR-WS, 2024, pp. 118–123.

[6] S.-W. Lee, M. Mohammadi, S. Rashidi, A. M. Rahmani, M. Masdari, M. Hosseinzadeh, et al., Towards secure intrusion detection systems using deep learning techniques: Comprehensive analysis and review, Journal of Network and Computer Applications 187 (2021) 103111.

[7] J. Lansky, S. Ali, M. Mohammadi, M. K. Majeed, S. H. T. Karim, S. Rashidi, M. Hosseinzadeh, A. M. Rahmani, Deep learning-based intrusion detection systems: a systematic review, IEEE Access 9 (2021) 101574–101599.

[8] P. Casas, J. Mazel, P. Owezarski, Unsupervised network intrusion detection systems: Detecting the unknown without knowledge, Computer Communications 35 (2012) 772–783.

[9] S. C. Tan, K. M. Ting, T. F. Liu, Fast anomaly detection for streaming data, in: Twenty-second international joint conference on artificial intelligence, Citeseer, 2011.

[10] P. A. A. Resende, A. C. Drummond, A survey of random forest based methods for intrusion detection systems, ACM Computing Surveys (CSUR) 51 (2018) 1–36.

[11] V. Agate, F. M. D'Anna, A. De Paola, P. Ferraro, G. Lo Re, M. Morana, A behavior-based intrusion detection system using ensemble learning techniques., in: ITASEC, 2022, pp. 207–218.

[12] M. Barcina-Blanco, J. L. Lobo, P. Garcia-Bringas, J. Del Ser, Managing the unknown in machine learning: Definitions, related areas, recent advances, and prospects, Neurocomputing (2024) 128073.

[13] J. Lu, A. Liu, F. Dong, F. Gu, J. Gama, G. Zhang, Learning under concept drift: A review, IEEE Transactions on Knowledge and Data Engineering 31 (2018) 2346–2363.

[14] J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, A. Bouchachia, A survey on concept drift adaptation, ACM computing surveys (CSUR) 46 (2014) 1–37.

[15] A. Augello, A. De Paola, G. Lo Re, M2fd: Mobile malware federated detection under concept drift, Computers & Security (2025) 104361.

[16] S. C. Hoi, D. Sahoo, J. Lu, P. Zhao, Online learning: A comprehensive survey, Neurocomputing 459 (2021) 249–289.

[17] F. Hinder, V. Vaquet, B. Hammer, One or two things we know about concept drift—a survey on monitoring in evolving environments. part a: detecting concept drift, Frontiers in Artificial Intelligence 7 (2024) 1330257.

[18] O. A. Wahab, Intrusion detection in the iot under data and concept drifts: Online deep learning approach, IEEE Internet of Things Journal 9 (2022) 19706–19716.

[19] V. Agate, S. Drago, P. Ferraro, G. Lo Re, Anomaly detection for reoccurring concept drift in smart environments, in: 2022 18th International Conference on Mobility, Sensing and Networking (MSN), IEEE, 2022, pp. 113–120.

[20] B. Celik, J. Vanschoren, Adaptation strategies for automated machine learning on evolving data, IEEE transactions on pattern analysis and machine intelligence 43 (2021) 3067–3078.

[21] R. Ade, P. Deshmukh, Methods for incremental learning: a survey, International Journal of Data Mining & Knowledge Management Process 3 (2013) 119.

[22] S. Grossberg, Nonlinear neural networks: Principles, mechanisms, and architectures, Neural

networks 1 (1988) 17–61.

[23] J. L. Lobo, J. Del Ser, M. N. Bilbao, C. Perfecto, S. Salcedo-Sanz, Dred: An evolutionary diversity generation method for concept drift adaptation in online learning environments, Applied Soft Computing 68 (2018) 693–709.

[24] R. Polikar, L. Upda, S. S. Upda, V. Honavar, Learn++: An incremental learning algorithm for supervised neural networks, IEEE transactions on systems, man, and cybernetics, part C (applications and reviews) 31 (2001) 497–508.

[25] R. Elwell, R. Polikar, Incremental learning of concept drift in nonstationary environments, IEEE transactions on neural networks 22 (2011) 1517–1531.

[26] A. De Paola, S. Drago, P. Ferraro, G. Lo Re, Detecting zero-day attacks under concept drift: An online unsupervised threat detection system, in: CEUR Workshop Proceedings, 8th Italian Conference on Cybersecurity, ITASEC, volume 2024, 2024.

[27] V. Agate, A. De Paola, S. Drago, P. Ferraro, G. Lo Re, Enhancing iot network security with concept drift-aware unsupervised threat detection, in: 2024 IEEE Symposium on Computers and Communications (ISCC), IEEE, 2024, pp. 1–6.

[28] G. Andresini, F. Pendlebury, F. Pierazzi, C. Loglisci, A. Appice, L. Cavallaro, Insomnia: Towards concept-drift robustness in network intrusion detection, in: Proceedings of the 14th ACM workshop on artificial intelligence and security, 2021, pp. 111–122.

[29] A. Tharwat, W. Schenck, A survey on active learning: State-of-the-art, practical challenges and research directions, Mathematics 11 (2023) 820.

[30] I. Žliobaitė, A. Bifet, B. Pfahringer, G. Holmes, Active learning with drifting streaming data, IEEE transactions on neural networks and learning systems 25 (2013) 27–39.

[31] T.-H. Lee, A. Ullah, R. Wang, Bootstrap aggregating and random forest, Macroeconomic forecasting in the era of big data: Theory and practice (2020) 389–429.

[32] Ž. Vujović, et al., Classification model evaluation metrics, International Journal of Advanced Computer Science and Applications 12 (2021) 599–606.

[33] A. Divekar, M. Parekh, V. Savla, R. Mishra, M. Shirole, Benchmarking datasets for anomaly-based network intrusion detection: Kdd cup 99 alternatives, in: 2018 IEEE 3rd international conference on computing, communication and security (ICCCS), IEEE, 2018, pp. 1–8.

[34] K. Siddique, Z. Akhtar, F. A. Khan, Y. Kim, Kdd cup 99 data sets: A perspective on the role of data sets in network intrusion detection research, Computer 52 (2019) 41–51.

[35] J. Vinagre, A. M. Jorge, C. Rocha, J. Gama, Statistically robust evaluation of stream-based recommender systems, IEEE Transactions on Knowledge and Data Engineering 33 (2019) 2971–2982.