

Continual Learning-Based Detection of Emerging Cyberattacks in Dynamic Network Environment

Prof. S. B. Bele¹, Kalyani R. Fande², Anandi D. Shelke³, Sharayu S. Pathak⁴

¹Professor, MCA Department,

^{2,3,4}MCA Department,

Vidya Bharati Mahavidyalaya, Amravati, Maharashtra, India

Abstract: Modern computer networks continuously change because of cloud services, IoT devices, mobile users, distributed applications, and evolving communication patterns. Conventional machine-learning intrusion detection systems are commonly trained on historical traffic and may degrade when traffic distributions or attack behaviors change. This paper proposes a Continual Learning-Based Emerging Cyberattack Detection Framework (CL-ECAF) for dynamic network environments. The framework integrates network-flow preprocessing, feature extraction, drift detection, incremental learning, experience replay, and knowledge-preservation mechanisms. The system continuously monitors incoming traffic, identifies distributional changes, and updates the detection model when new behavioral patterns appear. Replay samples help retain knowledge of earlier attacks, while regularization or knowledge distillation reduces catastrophic forgetting. The framework can be evaluated with temporally ordered CICIDS2017 and UNSW-NB15 traffic. Evaluation should include precision, recall, F1-score, false-positive rate, detection latency, adaptation time, forgetting, throughput, and memory usage. The proposed design provides a practical foundation for adaptive network intrusion detection capable of learning emerging threats without repeatedly rebuilding the complete model.

Keywords: Continual Learning, Cybersecurity, Network Intrusion Detection, Concept Drift, Emerging Attacks, Online Learning, Zero-Day Attacks, Deep Learning.

I. INTRODUCTION

The expansion of digital infrastructure has made computer networks essential to banking, healthcare, education, manufacturing, transportation, government services, and cloud computing. At the same time, the attack surface has increased. Distributed denial-of-service attacks, probing, botnets, credential attacks, ransomware, data exfiltration, and new attack variants can appear throughout the operational lifetime of a network.

Traditional intrusion detection systems depend on signatures, fixed rules, or models trained on historical data.

These methods can be effective for known patterns but are less suitable when legitimate behavior and attack strategies evolve. Such changes are commonly described as concept drift. In a dynamic network, drift can arise from new applications, software upgrades, cloud migration, changes in user behavior, new protocols, IoT deployment, or attacker adaptation.

Continual learning provides a way to update a model sequentially while attempting to retain previous knowledge. The major challenge is catastrophic forgetting, in which learning new information reduces performance on earlier knowledge. Replay, regularization, and knowledge-distillation methods have been proposed to balance stability and plasticity.

This paper proposes CL-ECAF, a framework that combines drift monitoring and continual learning for emerging cyberattack detection. The objective is not simply to maximize static classification accuracy, but to maintain detection quality over time while adapting to new traffic and attack behavior.



Objectives: (1) analyze limitations of static NIDS; (2) detect concept drift; (3) develop a continual adaptation mechanism; (4) reduce catastrophic forgetting; (5) support emerging-attack identification; and (6) define a comprehensive evaluation procedure.

II. LITERATURE REVIEW

Machine learning has become an important approach for intrusion detection. Decision trees, random forests, support vector machines, neural networks, and deep-learning models can learn relationships between traffic characteristics and attack classes. However, conventional supervised systems generally assume that training and testing distributions are reasonably stable.

Gama et al. studied concept-drift adaptation for evolving data streams and established a foundation for adaptive learning. In cybersecurity, drift may occur because of changes in legitimate behavior as well as changes in attack techniques. Recent IDS surveys emphasize drift detection, dynamic feature selection, adaptive learning, and continuous evaluation as important research areas.

Continual learning research generally groups methods into regularization, replay, and architectural approaches. Elastic Weight Consolidation protects parameters that are important to previous tasks. Learning without Forgetting transfers information from an existing model to a new learning stage. Experience replay retains representative historical examples and uses them during later updates. These techniques are relevant to NIDS because a security model must learn new threats without losing the ability to detect known attacks.

The research gap is that many NIDS studies still evaluate models using static random splits. Such evaluation can hide the effect of temporal change. In addition, detection accuracy alone does not measure adaptation speed, forgetting, memory cost, or operational false alarms. A continual NIDS should therefore be evaluated as a time-evolving system.

TABLE I
COMPARISON OF LEARNING STRATEGIES

Approach	Drift Handling	New Attacks	Forgetting Control	Adaptation
Signature IDS	Low	Low	N/A	Limited
Static ML	Limited	Limited	No	Low
Periodic Retraining	Yes	Yes	Moderate	Low
Online Learning	Yes	Yes	Limited	High
Continual Learning	Yes	Yes	Yes	High
Proposed CL-ECAF	Yes	Yes	Yes	High

III. PROPOSED METHODOLOGY

The proposed CL-ECAF architecture contains six stages: traffic collection, preprocessing and feature extraction, initial classification, concept-drift monitoring, continual adaptation, and security response. Incoming traffic is processed as a stream instead of being treated as one stationary dataset.



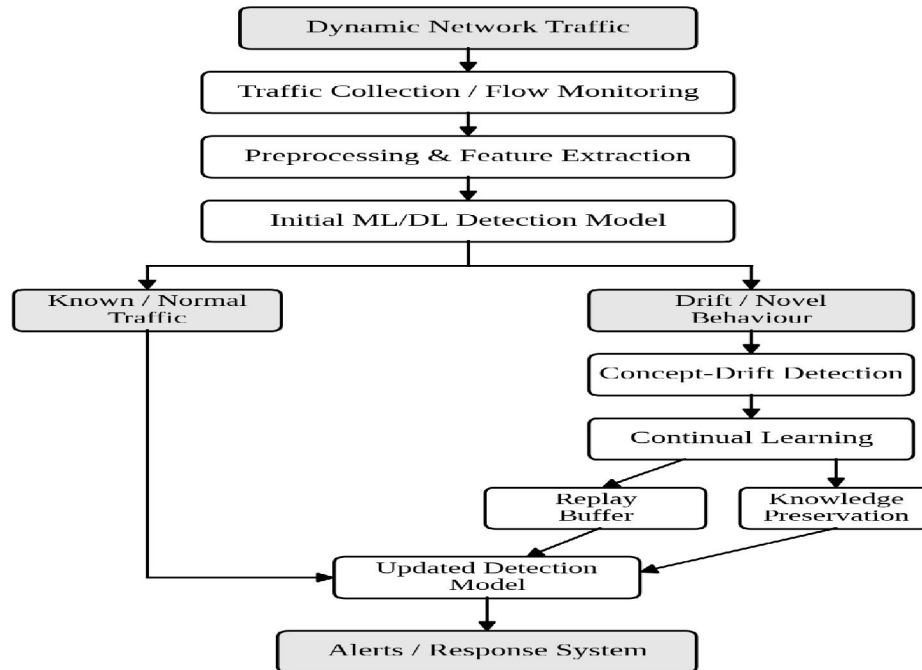


Fig. 1. CL-ECAF Architecture

A. Data Acquisition

The system receives network-flow records continuously. Useful attributes include source and destination ports, protocol, flow duration, packet and byte counts, packet-size statistics, inter-arrival times, TCP flags, and forward/backward traffic characteristics. CICIDS2017 and UNSW-NB15 can be used for experimental validation. To simulate deployment, records should be ordered chronologically rather than randomly shuffled.

B. Preprocessing

Preprocessing includes duplicate removal, missing-value treatment, categorical encoding, numerical scaling, class-imbalance handling, and chronological window creation. Identifiers that cause leakage should be excluded. A time-based split is preferred because the research question concerns performance on future traffic.

C. Feature Extraction

Let the traffic observation at time t be represented by $X_t = \{x_1, x_2, \dots, x_n\}$. Statistical and behavioral flow characteristics are extracted. Dynamic feature selection can be used to remove redundant attributes and retain features that remain useful under changing traffic distributions.

D. Initial Detection Model

The initial model may be implemented with a gradient-boosting classifier, multilayer neural network, or hybrid deep model. The classifier produces a probability distribution $P(y|X_t)$, where y is the predicted traffic class. A separate anomaly or uncertainty score can flag observations that do not resemble learned classes.

IV. CONTINUAL LEARNING AND DRIFT DETECTION

Let D_{old} represent the recent historical distribution and D_{new} represent the current traffic distribution. A drift score can be expressed as $S_{drift} = \text{Dist}(D_{old}, D_{new})$, where Dist may be implemented using a statistical test, divergence measure, density comparison, or streaming drift detector. If S_{drift} exceeds a threshold θ , the adaptation process is triggered.

When drift occurs, the updated learning set is $D_t = D_{new} \cup R_t$, where R_t is a set of representative samples from



the replay buffer. The updated loss can be represented as $L = L_{new} + \lambda_1 L_{replay} + \lambda_2 L_{preserve}$. L_{new} learns current traffic, L_{replay} maintains older examples, and $L_{preserve}$ protects previous model behavior.

The framework should use conservative adaptation. A drift event should not automatically be interpreted as a confirmed attack because legitimate network changes can also produce distribution shifts. New patterns can initially be assigned an uncertainty or emerging-behavior status and then be validated through security analysis.

Algorithm 1. Continual Learning-Based Detection

1. Initialize model M using historical traffic.
2. Initialize replay buffer R .
3. For each incoming window W :
4. Preprocess W and extract features.
5. Generate predictions using M .
6. Calculate drift statistics.
7. If $\text{drift} > \theta$:
8. Select representative current samples.
9. Retrieve samples from R .
10. Update M using continual-learning loss.
11. Apply knowledge-preservation mechanism.
12. Update R using memory policy.
13. Generate alerts for high-confidence malicious traffic.
14. Return updated M .

V. REPLAY BUFFER AND KNOWLEDGE PRESERVATION

The replay buffer should not contain only the most recent traffic. Such a policy could remove older attack categories and increase forgetting. A balanced buffer should retain normal traffic, known attacks, rare attacks, recent observations, drift samples, and uncertain cases.

Reservoir sampling can be used when memory is limited. Alternatively, importance-based selection can prioritize examples near decision boundaries or under-represented attack classes. The objective is to maximize the security value of stored examples rather than simply maximizing their number.

TABLE II

REPLAY BUFFER SAMPLE TYPES AND PURPOSES

Sample Type	Purpose
Normal traffic	Preserve legitimate behavior
Known attacks	Prevent forgetting
Rare attacks	Maintain minority-class knowledge
Recent traffic	Support adaptation
Drift samples	Capture new behavior
Uncertain samples	Improve decision boundaries

Knowledge preservation can be implemented with regularization or knowledge distillation. In an EWC-inspired approach, parameters that are important for previous knowledge receive larger penalties when changed. In a distillation approach, the previous model supplies soft targets that encourage the updated model to retain earlier decision behavior.

VI. EMERGING CYBERATTACK DETECTION

Emerging attacks may not have labeled examples in the initial training set. The framework therefore combines supervised classification with anomaly scoring, drift detection, and uncertainty estimation. A simple anomaly score can



be defined as $A(X) = 1 - \max_y P(y|X)$. High anomaly scores identify traffic that the classifier cannot confidently associate with known classes.

An observation can be prioritized as potential emerging behavior when it simultaneously exhibits high anomaly or uncertainty and participates in a significant distribution shift. This approach is preferable to treating every anomalous flow as an attack. Security analysts or secondary validation modules can investigate the resulting events.

The emerging-attack module can also use clustering to group unknown observations. If a cluster persists over multiple windows, it can be treated as a candidate new behavior and added to the continual-learning process after validation.

VII. EXPERIMENTAL DESIGN

The framework should be evaluated with a chronological stream. CICIDS2017 and UNSW-NB15 are appropriate starting points because they contain benign and multiple malicious traffic categories. A custom or real-world traffic stream should subsequently be considered for external validation.

TABLE III
DATASETS FOR EXPERIMENTAL EVALUATION

Dataset	Purpose	Evaluation
CICIDS2017	Modern network intrusion traffic	Chronological stream
UNSW-NB15	Diverse attack behavior	Chronological stream
Custom/real traffic	Deployment validation	Time-based evaluation

A useful evaluation sequence is: Stage 1 initial training; Stage 2 known traffic; Stage 3 distribution shift; Stage 4 new attack family or unknown behavior; Stage 5 continual adaptation; and Stage 6 long-term evaluation. This arrangement allows adaptation and forgetting to be measured directly.

VIII. PERFORMANCE METRICS

TABLE IV
PERFORMANCE METRICS AND THEIR PURPOSES

Metric	Purpose
Accuracy	Overall classification correctness
Precision	Correctness of predicted attack alerts
Recall	Ability to detect malicious traffic
F1-score	Balance between precision and recall
False Positive Rate	Operational false-alert burden
Detection Latency	Time to identify emerging behavior
Adaptation Time	Time needed to update the model
Forgetting	Loss of previous attack-detection capability
Memory Usage	Replay/model resource requirement
Throughput	Flows processed per unit time

For an imbalanced intrusion-detection problem, macro-F1, per-class recall, and false-positive rate should be emphasized rather than reporting accuracy alone. Performance should be measured before and after adaptation to determine whether the update improves new-threat detection without substantially reducing performance on older attacks.

IX. APPLICATIONS

- A. **Enterprise Networks:** The framework can monitor servers, endpoints, employee networks, and internal communication while adapting to changes in applications and user behavior.



- B. **Internet of Things:** IoT devices produce heterogeneous and evolving traffic. Continual learning can accommodate newly deployed devices and changing communication profiles.
- C. **Cloud Computing:** Cloud workloads change rapidly as virtual machines, containers, and services are created or removed. Adaptive detection can respond without full retraining.
- D. **5G and Future Networks:** High-speed networks generate large volumes of traffic and changing service patterns. Continual learning can support adaptive monitoring across network slices.
- E. **Smart Cities:** Transportation, energy, healthcare, surveillance, and communication systems create heterogeneous traffic, making continuous behavioral adaptation valuable.
- F. **Industrial IoT:** Industrial networks combine stable operational devices with newsensors and controllers. An adaptive IDS can help distinguish legitimate operational changes from suspicious deviations.
- G. **Emerging and Zero-Day Threats:** The framework can identify previously unseen patterns and prioritize them for investigation. Unknown behavior should be validated before becoming trusted training knowledge.

X. CHALLENGES AND SECURITY CONSIDERATIONS

Continual learning introduces new security risks. Attackers may attempt to poison the adaptation stream by inserting misleading observations. Therefore, automatic updates should be governed by confidence, drift persistence, validation rules, and potentially analyst approval.

Catastrophic forgetting remains a major challenge. Excessive adaptation can reduce performance on old attacks, while excessive regularization can prevent the system from learning genuinely new threats. The stability-plasticity trade-off must therefore be tuned experimentally.

Label scarcity is another challenge because emerging attacks may not be labeled immediately.

Semi-supervised, self-supervised, active-learning, and anomaly-detection methods can reduce dependence on immediate manual labeling. Computational cost, privacy, and false drift alarms are additional deployment concerns.

For critical environments, model updates should be versioned and reversible. A newly updated model should be validated before replacing a stable production model. This creates an operational safety layer against both accidental degradation and adversarial poisoning.

XI. FUTURE SCOPE

Future research can extend CL-ECAF toward federated continual learning, where multiple sites collaboratively improve detection without sharing raw traffic. Self-supervised representation learning can reduce the dependence on manually labeled attack examples. Explainable continual IDS models can provide analysts with reasons for predictions and explanations of why an update was triggered.

Adversarially robust continual learning is another important direction. Attackers may deliberately manipulate data streams or exploit the adaptation mechanism. Future systems should combine robust drift detection, adversarial training, secure model update policies, and rollback mechanisms.

Multimodal cybersecurity learning could combine network flows with endpoint logs, DNS events, authentication information, threat intelligence, and application telemetry. Automated replay-buffer management and lightweight edge-based adaptation are also promising areas for practical deployment.

Finally, human-in-the-loop continual learning can provide a useful compromise between full automation and static models. Security analysts can validate uncertain emerging patterns, after which confirmed knowledge can be incorporated into future model versions.

XII. CONCLUSION

Dynamic networks require intrusion detection systems that can learn continuously rather than relying exclusively on static historical training. The proposed CL-ECAF framework combines drift detection, incremental learning, replay, and knowledge preservation to address emerging cyberattacks while reducing catastrophic forgetting.



The framework emphasizes chronological evaluation and recommends measuring detection performance together with false-positive rate, adaptation latency, forgetting, throughput, and memory usage. This provides a more realistic assessment than static accuracy alone.

The proposed design is applicable to enterprise, IoT, cloud, 5G, smart-city, and industrial environments. Its main research value lies in treating intrusion detection as an evolving learning process. Future implementation and experiments should validate the framework under controlled temporal drift, emerging attack scenarios, adversarial conditions, and real network traffic.

REFERENCES

- [1]. J. Gama, I. Žliobaitė, A. Bifet, M. Pechenizkiy, and A. Bouchachia, "A Survey on Concept Drift Adaptation," *ACM Computing Surveys*, vol. 46, no. 4, 2014, doi: 10.1145/2523813.
- [2]. M. A. Shyaa, N. F. Ibrahim, Z. Zainol, R. Abdullah, M. Anbar, and L. Alzubaidi, "Evolving Cybersecurity Frontiers: A Comprehensive Survey on Concept Drift and Feature Dynamics Aware Machine and Deep Learning in Intrusion Detection Systems," *Engineering Applications of Artificial Intelligence*, vol. 137, 109143, 2024.
- [3]. M. Komarchesqui et al., "A Comprehensive Survey on Concept-Drift-Resilient Network Intrusion Detection Systems," *IEEE Access*, vol. 14, pp. 69689– 69718, 2026.
- [4]. L. Wang, X. Zhang, H. Su, and J. Zhu, "A Comprehensive Survey of Continual Learning: Theory, Method and Application," *arXiv:2302.00487*, 2023.
- [5]. J. Kirkpatrick et al., "Overcoming Catastrophic Forgetting in Neural Networks," *Proceedings of the National Academy of Sciences*, vol. 114, no. 13, pp. 3521– 3526, 2017.
- [6]. Z. Li and D. Hoiem, "Learning Without Forgetting," in *Proc. European Conference on Computer Vision*, 2016, pp. 614– 629.
- [7]. D. Rolnick, A. Ahuja, J. Schwarz, T. P. Lillicrap, and G. Wayne, "Experience Replay for Continual Learning," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [8]. D. J. Kavoi, A. Sahi, and C. J. Katila, "Deep Continual Learning Framework for Zero-Day Network Intrusion Detection Under Concept Drift," *SSRN*, 2026.
- [9]. M. Gama, J. Bifet, I. Žliobaitė, and M. Pechenizkiy, "Learning from Evolving Data Streams," in *Handbook of Big Data Technologies*, Springer, 2017.
- [10]. N. Moustafa and J. Slay, "UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems," *Proc. MilCIS*, 2015, pp. 1– 6.
- [11]. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization," *Proc. ICISSP*, 2018, pp. 108– 116.

