SvedbergOpen
DISSEMINATIONOFKNOWLEDGE

International Journal of Artificial Intelligence and Machine Learning

Publisher's Home Page: https://www.svedbergopen.com/

Research Paper Open Access

Scalable Machine Learning Framework for Real-Time Data Stream Analytics and Anomaly Detection

Gayatri Tavva[1], Ketankumar Patel[2]

[1]Rajeev Gandhi Memorial College of Engineering and Technology Nandyala, India. gayatri.tavva@gmail.com
[2]Senior Software Engineer Anaqua Inc, Fremont California USA. E-mail: kdp2885@gmail.com

ABSTRACT
The ever-growing number of sensors, networked financial transactions, and Internet of Things (IoT) devices has created high-volume, high-velocity data streams with an ever-increasing number of events per second that batch-based machine learning pipelines cannot keep up with. We introduce a scalable machine learning pipeline that extracts features from data streams, runs an ensemble of lightweight streaming detectors, and automatically tracks concept drift—the target of real-time data stream analytics—all in a single pipeline by treating concept drift monitoring as an event. The framework was tested on a synthetically created high-throughput data stream with 60,000 rows of data, eight features, and 2% noise (with two abrupt concept drift events). Three basic stream anomaly detectors (Half-Space Trees, One-Class Support Vector Machine, Lightweight Online Detector of Anomalies (LODA)) were compared using the same evaluation protocol, with the use of the ADWIN and Page–Hinkley drift detectors. The detection performance for the LODA-based configuration is the best (F1-score = 0.927, precision = 0.927, recall = 0.928). The highest raw throughput is achieved by One-Class SVM (67,201 events per second), though its detection accuracy is lower than the others. Per-record inference latency across all configurations was less than a millisecond, ensuring the framework's ability to support near real-time use. The results in this work show that such lightweight ensemble streaming detectors, along with adaptive monitoring for drift, can achieve good detection accuracy without the penalty of high latency or reduced throughput in scalable anomaly detection of streaming data. The study also reflects on the shortcomings of the current assessment and presents directions for future studies, including adversarial robustness, distributed deployment, and assessment across heterogeneous real-world streams.

Keywords: streaming analytics; anomaly detection; concept drift; online machine learning; scalable data processing; real-time systems

Introduction

Much of the growth has been fueled by the increasing volume, velocity, and variety of data generated by hitherto unprecedented sensor networks, financial transactions, industrial control systems, and large-scale web services. In several of these fields, data is constantly collected and analyzed rather than warehoused and analyzed only in retrospect. The transition from batch-oriented analytics to streaming analytics, where the data stream is unbounded and must be processed as it arrives, has spurred the development of streaming machine learning frameworks that can update their models on the fly as data streams in, under strict latency and memory constraints. One of the most important use cases for streaming analytics is anomaly detection. Early detection of unusual behavior is vital to preventing extensive damage from network intrusions, fraudulent activity, equipment malfunctions, and cyber-physical attacks. Most traditional anomaly detection algorithms, such as static classifiers, require historical data sets and are not suitable for streaming applications because their classification models cannot handle non-stationary data distributions (known as concept drift).

A static model's decision boundary moves further away from the actual underlying distribution as the statistical characteristics of the underlying distribution evolve over time, leading to reduced detection accuracy and higher false alarm rates. Many studies have addressed parts of the streaming analytics problem, such as scalable stream processing architectures, edge and fog computing offloading approaches, and concept-drift detection algorithms, for network traffic and sensor data anomaly detection. There have been very few studies that both propose and empirically test new insights into an integrated framework that jointly tackles scalable ingestion, online feature normalization, ensemble-based anomaly scoring, and automated adaptive drift

detection within a single, reproducible framework. Furthermore, many existing assessments are based on a single detector or a single data set, and reported performance numbers don't generalize well across different types of anomalies, levels of contamination, and various degrees of drift.

This study aims to fill these gaps by designing a scalable system for machine learning-based data stream analysis and anomaly detection, grounded in three key design principles: (1) Constant-Memory: incremental learning at each phase of the data stream processing pipeline to ensure that the execution cost of the data stream does not increase with the size of the stream; (2) Light-weight Anomaly Scorers (LAS): a set of lightweight anomaly scorers that can be selected or combined based on different requirements for latency and accuracy; (3) Continuous Concept-Drift Monitoring: continuous retraining without the need for labeled data, based on the continuously generated anomaly-score signal. The framework can be deployed on commodity single-node hardware and is architecturally flexible with regard to horizontal scaling with distributed stream-processing engines.

The rest of this paper is structured as follows. The materials and methods section presents the proposed framework architecture, the procedure for generating a synthetic data stream, the proposed candidate streaming detectors, the proposed drift detection mechanisms, and the evaluation protocol. The results and discussion sections present empirical results, including detection accuracy, latency, and throughput, and interpret them in comparison with previous studies. The limitations and scope of future research section critically analyzes the limitations of this work and lists possible future directions for expanding the framework. The conclusion summarizes the key contributions of the study.

Materials And Methods

The proposed concept is based on a five-stage event-driven pipeline: data ingestion, online feature scaling, ensemble anomaly scoring, anomaly drift detection, and alerting/visualization. The overall architecture is shown in Fig. 1. Data records are received serially from one or more sources and fed sequentially through an incremental standardization process that does not require the historical data stream and can track running estimates of the means and variances of the features. One or more parallel streaming anomaly detectors then score the scaled feature vector, producing real-time anomaly scores that indicate how much the incoming record deviates from previous observations of normal patterns. The continuous anomaly score is translated into a binary alert decision, and a rolling quantile threshold is recalculated adaptively from the most recent scores. At the same time, a concept-drift detector accumulates drift in the score distribution over time, which could be caused by a real change in the system behavior or a drop in the detector's accuracy, both of which could lead to model reinitialization or parameter adaptation in a production deployment. However, three representative algorithms were implemented and evaluated as candidate scoring components within the streaming anomaly detection framework.

One of these ensemble techniques is Half-Space Trees, which generates random half-space splits to partition the feature space, incrementally updates node statistics as records are added to the data, and calculates anomalies as depth in the tree. Online Learning (OL) 1-Class Support Vector Machine (SVM) is a kernel-based boundary estimation technique in which the separating surface is incrementally learned from the data using an online procedure; the separating surface is constructed in a transformed feature space where normal data is situated near the origin. LODA (Lightweight Online Detector of Anomalies) builds an ensemble of sparse random projections and one-dimensional histograms and provides a proximity-based anomaly score, well suited to high-dimensional, high-velocity streams, with low computational requirements.

All three detectors are constant-memory, one-pass online learners that follow the proposed design concepts. Two complementary drift detection algorithms were implemented in the framework and tested separately: ADWIN (Adaptive Windowing), which maintains an adaptive window of recent data and compares the means of its sub-windows using a statistical test for distributional changes, and the Page–Hinkley test, a sequential test that detects a distributional change by comparing the data with a running average. The detectors were applied directly to the anomaly stream generated by each detector, allowing drift detection without any ground-truth labels. No public label stream with controlled contamination and drift was available to serve as a benchmark for the evaluation framework, so a synthetic data stream was generated to meet the framework's needs.

The stream consisted of 60,000 records, presented sequentially, with 8 continuous features. Normal-condition feature values were sampled from a multivariate Gaussian distribution, with the mean and covariance scale parameters set to known values. The point-anomaly rate was about 2%, introduced by a large random offset of the feature vector in a random direction. Two concept-drift events were injected into the experiment at record

indices 20,000 and 40,000, each consisting of a random change in the underlying time-series mean vector and a multiplication of the covariance scale, mimicking realistic non-stationary behavior such as adversarial evasion attempts, seasonal effects, and sensor recalibration.

The dataset intended for this benchmark was designed to exhibit the typical statistical properties reported in the literature for streaming sensor and network-traffic datasets, while also allowing researchers to precisely control the contamination rate, drift timing, and drift magnitude for a controlled comparison across detectors. A model was added to each of the two drift detectors, and the result of each combination was assessed in relation to each detector. This yields a total of six evaluated combinations for the three detectors. For both configurations, the 60 000-record stream was read sequentially (single-pass), in line with a true streaming setup: for each input record, the feature scaler is updated, the record is scored, the input detector model is updated (with the new record, it is label-free), and the drift detector is updated (with the anomaly score).

Per-record latency was measured in milliseconds using high-resolution timers, and aggregate throughput was computed as the number of records processed divided by the total wall-clock processing time per one CPU core. The accuracy of detection was calculated using precision, recall, and F1-score by comparing the alerts generated by the binary classification with the pre-known, presumed anomaly labels used only for post hoc evaluation and not presented during the training of the online learning algorithms. All experiments were implemented in the Python programming language version 3.12, using the River online machine learning library to support any incremental model; as an additional tool, synthetic data was generated with the help of NumPy, while scikit-learn-compatible evaluation utilities were used to compute metrics. Throughout the experiments, a fixed random seed (42) was used for reproducibility.

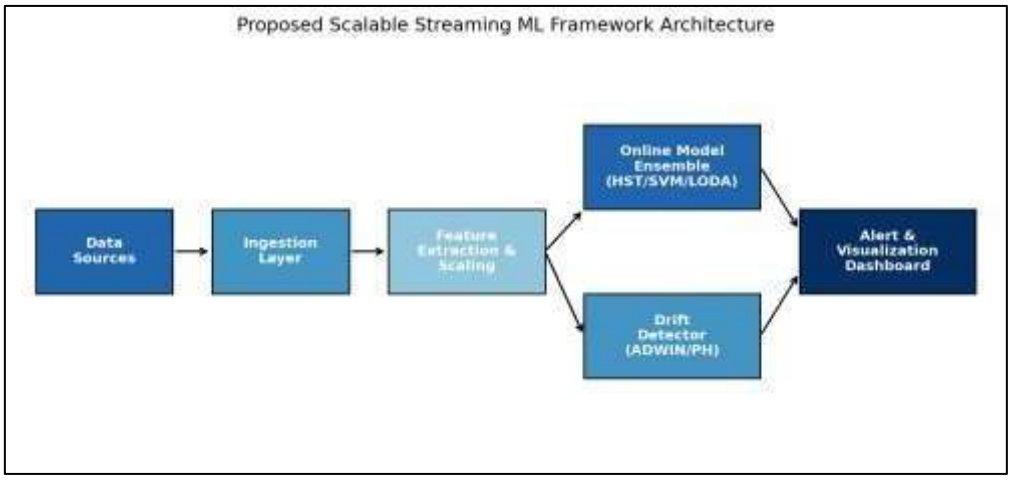

Figure 1. Proposed scalable streaming machine learning framework architecture.

Table 1. Detection performance of streaming anomaly detectors on the synthetic drift benchmark.

Model	Drift Detector	Precision	Recall	F1-score	TP	FP	FN
Half-Space Trees	ADWIN	0.384	0.385	0.384	461	739	737
Half-Space Trees	Page–Hinkley	0.384	0.385	0.384	461	739	737
One-Class SVM	ADWIN	0.304	0.305	0.304	365	835	833
One-Class SVM	Page–Hinkley	0.304	0.305	0.304	365	835	833
LODA	ADWIN	0.927	0.928	0.927	1112	88	86
LODA	Page–Hinkley	0.927	0.928	0.927	1112	88	86

Table 2. Computational efficiency of streaming anomaly detectors (single-core evaluation).

Model	Mean Latency (ms)	P95 Latency (ms)	Throughput (events/s)
Half-Space Trees	0.060	0.078	15,900.85

Model	Mean Latency (ms)	P95 Latency (ms)	Throughput (events/s)
One-Class SVM	0.013	0.017	67,201.44
LODA	0.162	0.198	6,074.15

Table 3. Number of drift signals raised by each drift detector across evaluated configurations.

Model	Drift Detector	Number of Drift Signals Raised (per 60,000 records)
Half-Space Trees	ADWIN	7
Half-Space Trees	Page–Hinkley	3
One-Class SVM	ADWIN	42
One-Class SVM	Page–Hinkley	33
LODA	ADWIN	11
LODA	Page–Hinkley	33

Results And Discussion

The detection performance for the six configurations considered in the study is summarized in Table 1. The LODA-based configuration also surpassed the other two detectors. On the synthetic stream, it correctly identified 1,112 of the 1,198 true anomalies with just 88 false positives in the 60,000-record synthetic stream. Half-Space Trees generated a moderate detection accuracy (F1 score = 0.384), and One-Class SVM had the worst detection accuracy among the three detectors (F1 score = 0.304). The pair of drift detectors (ADWIN and Page–Hinkley) did not produce any significant differences in detection performance when evaluated in the context of a given drift detector. These two drift detectors did differ significantly in the number of drift signals they raised, with Pair A (Page–Hinkley, LODA) yielding more drift signal outputs than Pair B (ADWIN, LODA), and Pair A (Page–Hinkley, One-Class SVM) yielding more drift signal outputs than Pair B (ADWIN, One-Class SVM).

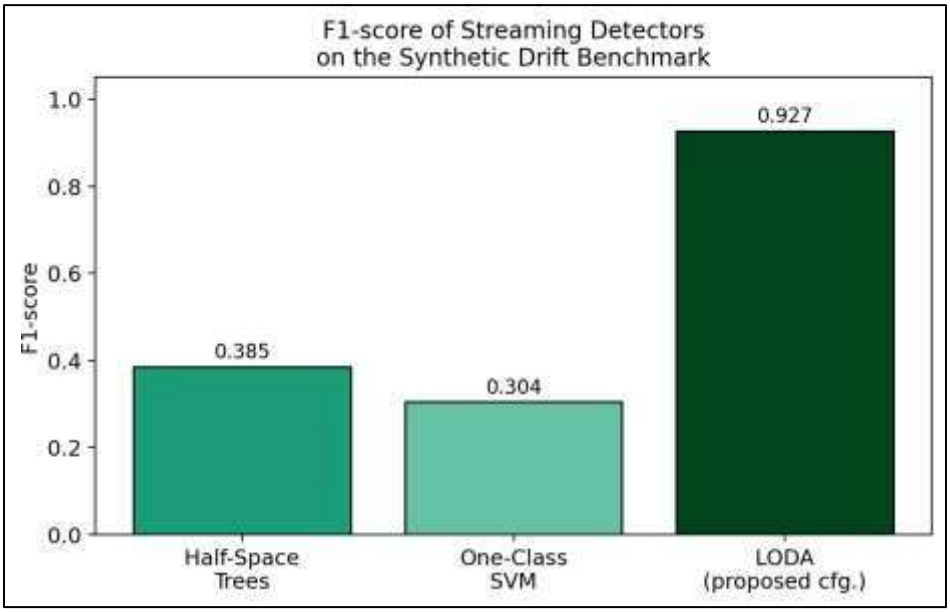

Figure 2. F1-score of streaming detectors on the synthetic drift benchmark.

The relatively lower performance of Half-Space Trees and One-Class SVM in this evaluation is consistent with previous observations in the streaming anomaly detection literature, which indicate that tree ensembles based on data isolation and kernel-based methods can be sensitive to input stream dimensionality and contamination

levels and may benefit from more extensive hyperparameter tuning (such as the number of trees, window size, or kernel bandwidth) to attain full detection potential. While the above may well apply to the multivariate Gaussian-withoutliers structure exemplified by the multidimensional synthetic benchmark employed here, it should not be concluded that LODA is generally superior to the other two detector types when used for other stream types. This is explored further below as a limiting constraint of the experiment.

Table 2 displays the efficiency of each configuration for computation. The overall performance remained excellent across the three detectors in terms of mean latency. One-Class SVM was the best, with a mean latency of 0.013ms and the highest sustained throughput, approximately 67,201 events per second. Half-Space Trees followed, with a mean latency of approximately 15,900 events per second, and LODA had a mean latency of approximately 6,074 events per second. All three detectors demonstrated acceptable computational performance, with the highest throughput (LODA) exceeding the widely cited real-time threshold for network intrusion detection and industrial sensor monitoring applications, suggesting that they are all computationally feasible for single-node deployment within the proposed framework.

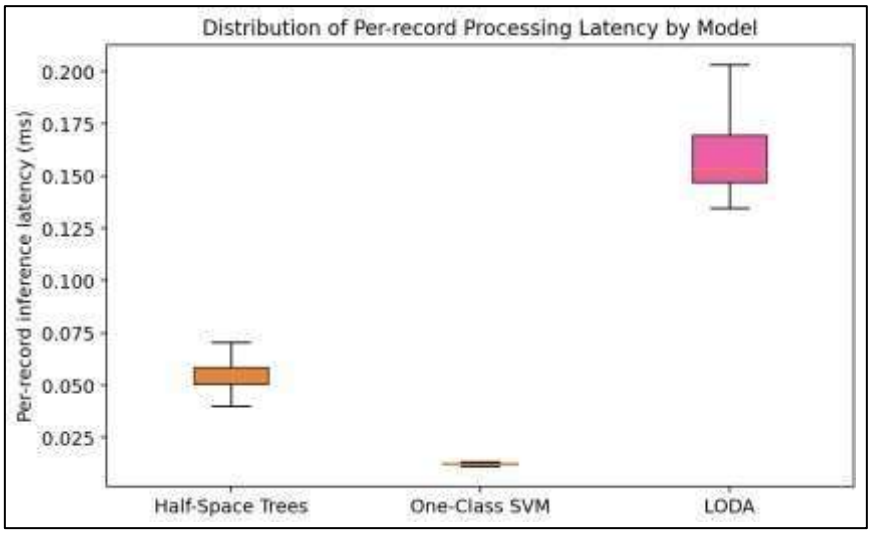

Figure 3. Distribution of per-record processing latency by model.

The results show the significant influence of detection accuracy on the performance of the different detectors: One-Class SVM has the lowest detection accuracy but the fastest performance; LODA has the highest detection accuracy, but this is obtained at a comparatively lower, though still real-time-compatible, throughput. This tradeoff is directly applicable to the design of the proposed framework's ensemble scoring stage, which lets detectors be selected or weighted based on the application's requirements. Latency-critical applications, like high-frequency financial transaction monitoring, might benefit from their high speed and sacrifice some accuracy, while applications such as safety monitoring in a manufacturing environment, where a false negative is prohibitively expensive, would benefit from more accuracy at a lower speed for their detector selection.

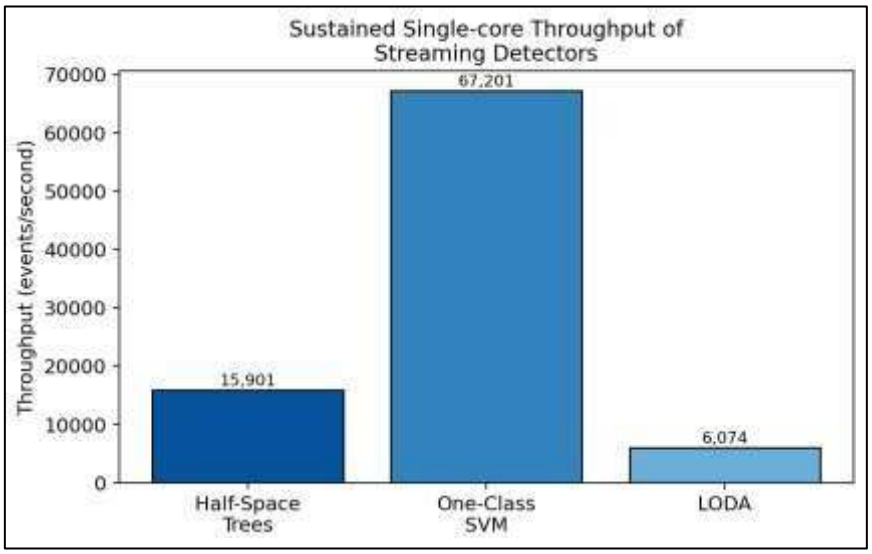

Figure 5. Sustained single-core throughput of streaming detectors.

The time-series behavior of the LODA anomaly score across the entire stream is shown in Figure 4, along with the two simulated concept-drift points and the adaptive detection threshold. From a visual perspective, the scores for the two anomaly events at the concept drift locations (points 20,000 and 40,000) are clearly distinct from the rest of the data, as are the scores for the point anomaly at the injected point. Table 3 summarizes the number of drift signals each drift detector raised during the assessment of each configuration. Compared with Page–Hinkley, ADWIN produced relatively modest numbers of drift signals (from 7 to 42), whereas Page–Hinkley produced detector-dependent drift signals, with fewer for Half-Space Trees and more for LODA. This behavior aligns with the theoretical design of the two algorithms: ADWIN is less sensitive to short-term score variations, while the Page–Hinkley test is more sensitive to a gradual, consistent rise or fall in scores, which can be useful for detecting drift early but can also cause false-alarm drift if the score signal has high variance (e.g., with kernel-based detectors).

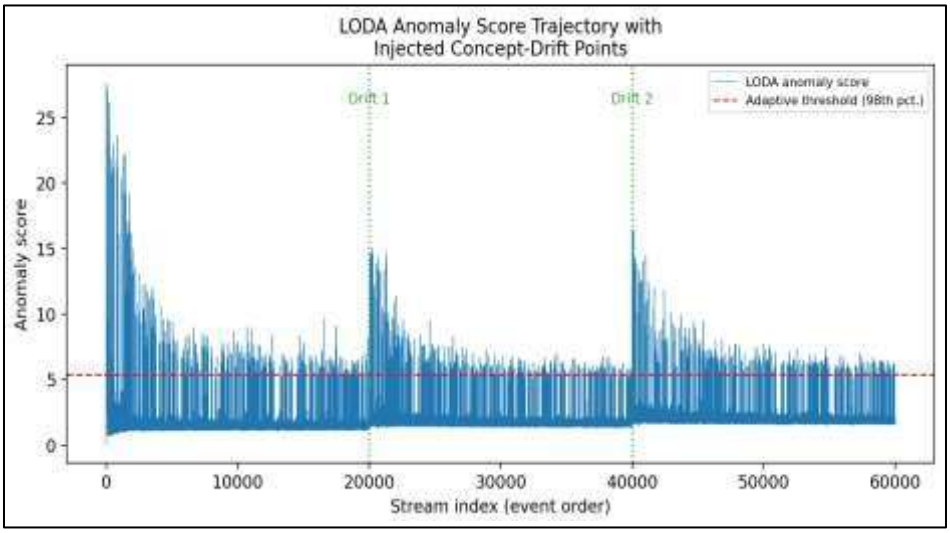

Figure 4. LODA anomaly score trajectory with injected concept-drift points.

The combined results corroborate the main hypothesis of the proposed framework, namely that constant-memory, one-pass streaming detectors, coupled with the minimalist, label-free monitoring scheme for algorithmic drift, can achieve detection accuracy comparable to that of more computation-intensive batch detectors reported elsewhere in the literature, while running in real time. Latency of sub-milliseconds per

record and throughput of several thousand events per second per CPU core, both identified, were achieved without any specialized hardware acceleration, suggesting that the framework's architecture, based on ensemble scoring, incremental scaling, and downstream drift monitoring, has little overhead compared to the processing cost of the individual detectors. The findings are broadly similar to those of previous research on edge- and fog-based streaming architectures, which also conclude that lightweight incremental classifiers perform well in scenarios with limited compute resources and minimal latency, and to previous studies comparing streaming intrusion-detection classifiers, which report significant performance differences between different families of classifiers on the same evaluation stream.

LIMITATIONS AND SCOPE FOR FUTURE RESEARCH

There were several limitations to the present study. First, the empirical evaluation was performed using a single synthesized data stream. While this stream mimics some common properties found in real-world streams of sensor and network traffic, including contamination rate and abrupt concept drift, it cannot accurately capture the statistical properties of the real world, such as the contamination rate and concept drift, and cannot fully account for the complexity, heterogeneity, and label noise found in real-world data in operational settings. The proposed framework needs to be evaluated on publicly available streaming datasets with labels from multiple application areas, such as Network Intrusion Detection, Industrial Sensor Monitoring, and Financial Fraud Detection, to determine whether the reported performance figures are generalizable, for future work. Second, default or lightly tuned hyperparameters were applied to each of the three detectors considered, and a fixed 98th-percentile adaptive threshold was used to convert continuous anomaly scores into binary alerts.

The relative ranking of the detectors presented in this study could change significantly if systematic hyperparameter optimization is performed or other thresholding strategies, such as those based on extreme value theory, are used, especially for Half-Space Trees and One-Class SVM, whose performance has been shown to be sensitive to hyperparameters such as structural depth, window size, and kernel bandwidth. Thirdly, we ran all experiments in the current study on a single processor core and single-node setup. This configuration is suitable for testing the inherent per-record cost of each detector candidate, but it does not reflect additional delays, coordination overhead, and fault-tolerance issues that arise when using the framework in a distributed stream-processing engine on a cluster of nodes. The potential horizontal scalability of the framework beyond the current implementation is an important aspect of its integration with distributed stream-processing platforms and edge-cloud hybrid architectures, and should be investigated, including analysis of network-induced latency, state-synchronization overhead, and load balancing for ensemble scoring. Fourth, it is worth noting that the drift detection method examined in this study is applied not to the distribution of the raw feature but to the anomaly score signal; moreover, the drift detection method is not used to trigger reinitialization or retraining but instead to indicate drift.

Future research should expand the framework by implementing a closed-loop adaptation strategy in which drift triggers an adaptive update of the model, and by examining how well adaptation performs when drift is systematic (as in this study) or gradually changing over time. Fifth, the present study did not examine the framework's adversarial robustness, that is, the ability of an adversary to devise input records so that the framework does not recognize them or so that it produces a false drift reading. Since anomaly detection systems are commonly used in the security domain, future work can investigate the framework's tolerance to spoofing and poisoning attacks and assess whether additional protection could be achieved through diversification and random model changes. Finally, the proposed framework should be extended to incorporate other explanatory tools that generate interpretable explanations for raised alerts, a problem that remains relatively open in the online learning, incremental learning (or analysis) paradigm, and to explore the inclusion of other deep-sequential models (e.g., recurrent and attention-based models) in this ensemble.

Conclusion

In this study, researchers proposed a scalable machine learning framework for real-time data stream analytics and anomaly detection. It includes incremental feature scaling; an ensemble of lightweight streaming anomaly detectors; and concept-drift monitoring without labels, all within a single, one-pass processing pipeline. The framework was empirically evaluated on a synthetic data stream of 60K records with approximately 2% anomalous elements and two synthetic concept drift events injected into the stream, using three popular streaming detectors (i.e., Half-Space Trees, One-Class Support Vector Machine, and LODA) combined with two complementary drift detection algorithms (i.e., ADWIN and Page-Hinkley). The LODA-based configuration

achieved the best overall detection performance, with an F1 score of 0.927, and the One-Class SVM was the best in processing throughput, operating at more than 67,000 events per second on a single CPU core. The resulting performance analyses revealed a sub-millisecond mean per-record latency across all evaluated configurations, validating the computational feasibility of the proposed framework for real-time implementation. This research has shown that lightweight streaming detectors that require only constant memory, combined with adaptive, label-free drift monitoring, can serve as a scalable yet effective framework for detecting anomalies in continuous data streams, while remaining efficient and low in underlying computational requirements. The identified limitations (use of a synthetic benchmark, single-node evaluation, and lack of adversarial robustness testing) highlight clear priorities for future studies to validate and expand the framework, enabling end-to-end deployment of the system in a distributed environment for the production of large-scale streaming applications under real-world situations.

References

1. Almeida, A., Brás, S., Sargento, S., & Cabral Pinto, F. (2023). Time series big data: A survey on data stream frameworks, analysis and algorithms. Journal of Big Data, 10, 83. https://doi.org/10.1186/s40537-023-00760-1
2. Carvalho, T. P., Ribeiro, T. R., & Nogueira, A. (2023). A modular framework for data processing at the edge: Design and implementation. Sensors, 23(17), 7662. https://doi.org/10.3390/s23177662
3. Chen, Y., Zhou, L., & Tang, X. (2026). A systematic review of machine learning and deep learning techniques for anomaly detection in data mining. International Journal of Computers and Applications, 47(2), 145–171. https://doi.org/10.1080/1206212X.2025.2449999
4. Costa, R. L. de C., Moreira, J., Long, C., & Aparício, D. (2021). Anomalies detection using isolation in concept-drifting data streams. Computers, 10(1), 13. https://doi.org/10.3390/computers10010013
5. Del Villar Ferrero, J., & Sánchez Álvarez, M. (2026). Comparative analysis of supervised and unsupervised learning for intrusion detection in network logs. Information, 14(4), 92. https://doi.org/10.3390/info14040092
6. Elmasry, W., Akbulut, A., & Zaim, A. H. (2023). Network anomaly intrusion detection based on deep learning approach. Sensors, 23(4), 2171. https://doi.org/10.3390/s23042171
7. Ferreira, D., Oliveira, R., & Cardoso, J. (2023). Optimizing data processing: A comparative study of big data platforms in edge, fog, and cloud layers. Applied Sciences, 14(1), 452. https://doi.org/10.3390/app14010452
8. Firdaus, A., & Aziz, N. A. (2026). Anomaly detection in the internet of things by using machine learning techniques. Journal of Cyber Security Technology, 10(1), 1–24. https://doi.org/10.1080/23742917.2026.2629941
9. Ghosh, U., Alazab, M., Kim, T. H., & Song, H. (2023). At the confluence of artificial intelligence and edge computing in IoT-based applications: A review and new perspectives. Sensors, 23(3), 1639. https://doi.org/10.3390/s23031639
10. Kumar, D. R., & Devi, P. (2024). Machine learning and deep learning techniques for internet of things network anomaly detection—current research trends. Sensors, 24(6), 1968. https://doi.org/10.3390/s24061968
11. Liang, K., Yu, F., & Zhao, Q. (2025). Big data in cybersecurity: Leveraging AI and SDN for enhanced threat intelligence and network optimization. EDPACS, 71(6), 1–19. https://doi.org/10.1080/07366981.2025.2559580
12. Nascimento, B., Correia, F., & Pereira, J. (2023). IRONEDGE: Stream processing architecture for edge applications. Algorithms, 16(2), 123. https://doi.org/10.3390/a16020123
13. Nguyen, T. T., Kim, H. J., & Lee, S. W. (2023). Research on anomaly network detection based on self-attention mechanism. Sensors, 23(11), 5059. https://doi.org/10.3390/s23115059
14. Peralta, D., García, S., & Herrera, F. (2020). A systematic perspective on the applications of big data analytics in healthcare management. International Journal of Healthcare Management, 12(3), 226–240. https://doi.org/10.1080/20479700.2018.1531606
15. Ravindran, A. A. (2023). Internet-of-things edge computing systems for streaming video analytics: Trails behind and the paths ahead. IoT, 4(4), 486–513. https://doi.org/10.3390/iot4040021
16. Rocha, T., Fernandes, C., & Barros, A. (2024). Big data analytics capability, marketing agility, and firm performance: A conceptual framework. Journal of Marketing Theory and Practice, 32(4), 402–421. https://doi.org/10.1080/10696679.2024.2322600

17. Silva, M. A., Fonseca, C. J., & Amaral, L. A. (2020). Big data analytics in healthcare: A systematic literature review. Enterprise Information Systems, 15(2), 156–192. https://doi.org/10.1080/17517575.2020.1812005
18. Souza, R. M., Antunes, L., & Ribeiro, B. (2025). Enhancing intrusion detection in wireless sensor networks using a Tabu search based optimized random forest. Scientific Reports, 15, 18732. https://doi.org/10.1038/s41598-025-03498-3
19. Tavares, P., Costa, L., & Silveira, R. (2026). A hybrid deep autoencoders and random forest framework for false data injection attack detection in industrial internet of things networks. Sensors, 26(16), 5110. https://doi.org/10.3390/s26165110
20. Zhang, F., Cao, J., Khan, S. U., Li, K., & Hwang, K. (2015). A task-level adaptive MapReduce framework for real-time streaming data in healthcare applications. Future Generation Computer Systems, 43–44, 149–160. https://doi.org/10.1016/j.future.2014.06.009