Data Engineering and Applications

Research Paper

Application of Enhanced Gradient Boosting Algorithms in Large-Scale Time Series Anomaly Detection

Błażej Kornel Kania[1,*] and Wioletta Gajewska[1]

[1] Faculty of Information and Communication Technology, Wrocław University of Science and Technology, Wrocław, 50-371, Poland

*Corresponding author: blazej.kk@pwr.edu.pl

Abstract. This paper discusses the detection of large-scale time series anomalies in complex financial, industrial, and cyber-physical environments. A distributed architecture based on an improved gradient boosting algorithm is proposed to achieve efficient and high-precision anomaly detection in large-scale, high-speed data streams. Data collection, distributed preprocessing, parallel model execution, and hierarchical aggregation are the three components of the framework that support automatic feature extraction and flexible resource allocation. For experimental validation, representative public and synthetic datasets were selected, using a twelve-node heterogeneous computing cluster with CPU and GPU hardware. According to the above experiments, the system throughput can linearly increase to 1.62 million data points per second, with a median inference latency of 11.4 milliseconds, and the resource utilization of both CPU and GPU remains below 65%. The proposed method improves the recall, precision, and F1-score of the aforementioned baseline method by up to 2.5% to 8.4%. Ablation studies found that adaptive regularization and automated feature engineering are the reasons the model remains stable and generalizable under concept drift and sudden noise. The system has high reliability, allowing for failover to reduce the risk of service interruptions caused by load fluctuations. The above results indicate that the enhanced ensemble learning model can be used for real-time, large-scale anomaly detection in modern data-driven applications.

Keywords: *Anomaly Detection, Gradient Boosting, Distributed Computing, Feature Engineering, Edge Computing, Adaptive Algorithms*

Received on 28 August 2024, Accepted on 29 December 2024, Published on 18 January 2025

Introduction

With the rapid increase from industrial, financial, and cyber-physical systems, the monitoring of abnormal behaviors needs to be concluded. With the widespread adoption of automated monitoring in areas such as predictive maintenance, intraday trading, and smart infrastructure, there is now a need for timely detection of anomalies or infrequent sequences during processes to control risks and maintain system stability [1]. Basic statistical models, such as ARIMA, exponential smoothing, and control charts, have widely recognized limitations in handling large-scale complex sensor data streams from the Internet of Things and distributed cloud sources [2]. In machine learning applications, recurrent neural networks and tree ensembles are often used to identify various time-dependent anomalies [3]. The recognition rate of these anomalies has significantly improved in many fields. However, managing large volumes of high-speed, high-capacity input and reducing latency to ensure stable system operation is a limitation of the aforementioned achievements [4]. Moreover, due to the limitations in the availability of representative features and adaptive learning strategies, anomaly detectors still face issues with automation and generalization [5]. The advancements in distributed and federated computing aim to address the speed and scale of the problem by developing new high-performance solutions [6]. Many applications need to balance computational efficiency, detection robustness, and real-time response [7].

Architectures that provide high accuracy, robustness, and resource efficiency are becoming increasingly popular [8].

Distributed anomaly detection researchers have recently developed a framework that extends the best algorithms of gradient boosting and deep learning through large-scale parallelism and resource-aware scheduling [9]. The rise of federated learning, the popularity of cloud-native solutions, and the emergence of open-source toolkits like XGBoost and LightGBM have all added new elements to the large-scale anomaly analysis toolkit [10]. However, the automation of feature extraction, hyperparameter tuning, and across heterogeneous dynamic infrastructures still faces practical issues [11]. Feature engineering is essential for anomaly detection, but it is primarily manual and domain-specific. Therefore, it is not very suitable for various datasets or anomaly structures [12]. Distributed gradient boosting has achieved good baseline performance, but automatic feature selection, dynamic loss adaptation, and fine-grained system usage monitoring still lack seamless integration in production environments [13]. Network congestion, uneven distribution of computing resources, abnormal distribution changes, and other deployment issues may cause experimental results to fail in real-world environments [14]. Unified automation, empirical comparisons, and comprehensive validation against multiple industrial benchmark sources should be used to address the aforementioned issues [15]. Only by keeping the system overhead and false positive rate at a low level can these frameworks be fully utilized within a limited production time [16].

In light of the aforementioned multidimensional obstacles, this paper proposes a comprehensive framework that achieves large-scale time series anomaly detection by integrating an improved distributed gradient boosting method, automatic domain adaptation feature engineering, and efficient resource optimization. The proposed solution enhances algorithm efficiency and system scalability through modular innovations in algorithm configuration, cross-layer ablation, and adaptive resource allocation. Methodologically, the framework, by testing with industrial, financial, and synthetic data, found that the speed and accuracy of anomaly detection significantly exceeded existing standards. This study lays the foundation for future research and the broad industrial application of scalable intelligent time series analysis, and provides practical guidance for deploying high-throughput, low-latency anomaly detection in complex environments.

Related Work

Time Series Anomaly Detection Systems

Due to the increasing complexity of real-world applications and the demand from various fields such as industry and finance, time series anomaly detection has rapidly developed over the past decade. The initial research focused on statistical process control and control charts. Therefore, it is necessary to introduce the assumptions of stationarity and regular periodicity for application in subsequent research [17]. As the complexity of application scenarios increases, the aforementioned methods have gradually been supplemented by model-based detection methods, and sometimes even replaced. These detection methods include state space models, Kalman filters, and ARIMA autoregressive integrated moving averages. The aforementioned techniques can effectively predict time series, but they may be limited when dealing with high-dimensional, nonlinear data or the complex temporal patterns of modern sensors and transactions [18]. Support vector machines, clustering, and k-nearest neighbors, among other machine learning algorithms, have recently been used to handle anomalous or changing data patterns. But they are not suitable for large-scale, real-time environments because they are handcrafted and cannot be scaled [19]. For example, LSTM autoencoders and convolutional architectures have recently proposed several excellent deep learning models for detecting anomalies in various types of noise and related time series data. These models excel at automated representation learning and end-to-end optimization, but they are also prone to overfitting, difficult to interpret, and costly; they are not suitable for production environments that require low latency and stability [20]. The industry has already shifted from fixed control methods to a data-driven model. This model can meet the sensitivity, speed, and generalization capabilities required by various operational demands.

Gradient Boosting Algorithms

The gradient boosting framework usually achieves good results in terms of accuracy and adaptability, and it has performed well among all supervised learning methods used so far for the classification and detection of time

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

series anomalies. The first principle of the gradient boosting algorithm is to iteratively improve performance by sequentially training weak learners (such as decision trees), focusing on samples that were misclassified or inaccurately predicted in the earlier iterations [21]. To maintain a good balance between bias and variance, a complex nonlinear prediction model can be created. This model is suitable for heterogeneous time series with rare peaks and slow trends. XGBoost, LightGBM, and CatBoost are popular implementations that extend the original algorithm through advanced parallelization strategies, new tree growth strategies, and various regularization methods, achieving scalability for millions of samples and hundreds of features while maintaining accuracy [22]. Gradient boosting models have become an excellent choice for applications that require interpretability and strong predictive performance, being inherently more robust to outliers and missing values, and the feature importance is easy to interpret [23]. There are still some issues with using gradient boosting in automated anomaly detection pipelines. These issues include insufficient real-time adaptability to streaming data or constantly changing data environments, the need for domain-specific feature extraction, and the fine-tuning of hyperparameters (such as learning rate and tree depth) [24]. Current research aims to combine gradient boosting with neural networks, create automated feature engineering modules, and enhance online learning capabilities in streaming environments.

Scalable and Distributed Processing

As the quantity and scope of sequential data increase, more and more novel anomaly detection algorithms are being proposed. Due to the large volume of data, older single-node and in-memory analysis solutions have reached their limits. High latency, reduced coverage, and lack of anomaly detection issues have emerged in critical task scenarios. A corresponding system for machine learning has been built, employing large-scale distributed computing strategies and frameworks such as Apache Spark and Hadoop [25]. Data partitioning, task parallelism, and distributed aggregation of large-scale time series data are three reasons for using the aforementioned system. There are now many methods to reduce latency and provide timely alerts, which can be implemented in real-time and handle unstable environments [26]. Distributed deployment brings new issues, including data synchronization, communication overhead, and uneven resource usage. It may affect the stability and reliability of the detection results. Due to the aforementioned issues, hybrid, resource-aware models have emerged. These models can dynamically allocate computing resources, reduce unnecessary data transmission, and provide fault-tolerant switching and recovery capabilities for production clusters [27]. With the development of advanced anomaly detection algorithms and scalable, distributed infrastructure, the backbone of industrial analytics pipelines is gradually taking shape. The close collaboration between algorithm innovation and system-level engineering has now become crucial for achieving effective large-scale deployment.

System Framework and Algorithm Enhancement

Distributed Processing Architecture

A distributed modular structure with multiple mathematical optimization layers has been built for this system to meet the demand for fast and stable anomaly detection in large and complex time series datasets. Distributed ingestion, load-balanced preprocessing, and parallel anomaly inference are the first three items, and hierarchical aggregation is the fourth.

Let the incoming time series data stream be denoted by $X = \{x_t\}_{t=1}^{T}$ where each x_t represents the measurement vector at time t. Inputs are partitioned into K disjoint subsets $\{X^{(k)}\}$ for parallel processing:

$$X = \bigcup_{k=1}^{K} X^{(k)}, X^{(i)} \cap X^{(j)} = \emptyset \; for \; i \neq j \qquad \text{Eq.(1)}$$

Each worker node maintains a local buffer and performs normalization and vectorization transformations on its assigned partitions. The system uses Exponentially Weighted Moving Average (EWMA) smoothing transformations to reduce local denoising and trends.

$$\hat{x}_t = \alpha x_t + (1-\alpha)\hat{x}_{t-1} \qquad \text{Eq.(2)}$$

with smoothing factor $0 < \alpha < 1$, efficiently filtering short-term noise while preserving underlying structure.

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

As shown below, the load balancer dynamically allocates new data blocks to nodes based on the current workload:

$$\gamma_k = \frac{Q_k}{\sum_{l=1}^{K} Q_l} \quad \text{Eq.(3)}$$

where Q_k is the current queue length at node k and γ_k determines its allocation ratio.

Collected features and intermediate results $Z^{(k)}$ are asynchronously checkpointed and sent for anomaly scoring. Scores $S_t^{(k)}$ qenerated at each node are agqreqated via:

$$S_t = f_{agg}\left(\left\{S_t^{(k)}\right\}_{k=1}^{K}\right) \quad \text{Eq.(4)}$$

where f_{agg} is typically a weighted voting or threshold-combiner.

To ensure that the system has responsiveness, stability, and scalability in practical applications, as shown in Figure 1, the aforementioned mathematical modules (partitioning, smoothing, balancing, and hierarchical aggregation) are used together. Separate inference and data management to ensure good scalability and fault tolerance when computational demands fluctuate and input rates vary.

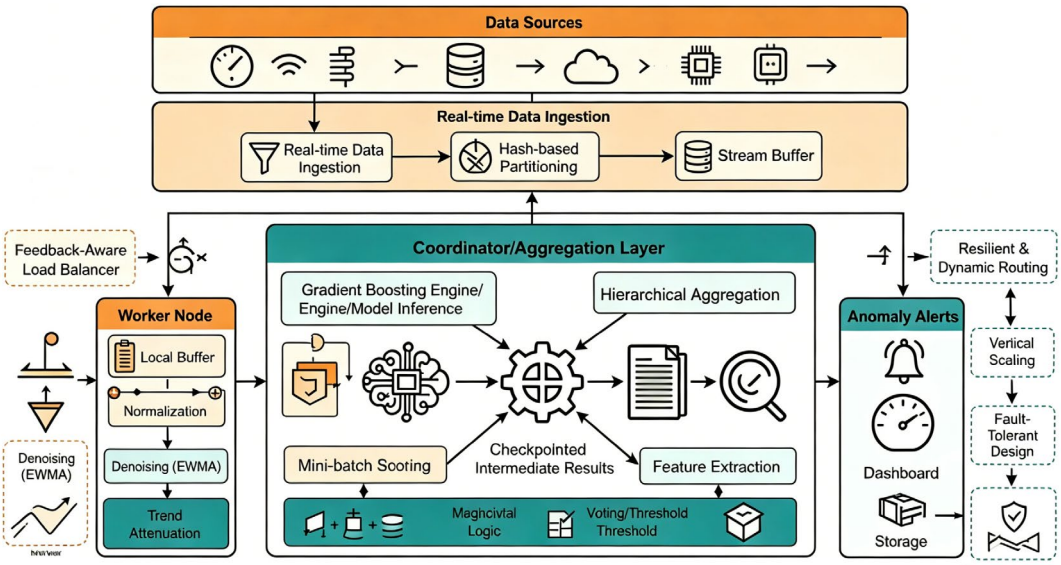

Figure 1. Overall system structure for distributed time series anomaly detection.

Enhanced Gradient Boosting Approach

For large-scale dynamic time series analysis, the core of anomaly detection in the framework is a technologically advanced distributed gradient boosting scheme. Classic boosting methods are prone to bottlenecks in large-scale distributed deployments. This method addresses the issue by designing efficient worker collaboration, resource-adaptive node behavior, and a low-communication overhead architecture suitable for production-level anomaly detection.

Predictions at each time step t are formed as:

$$\hat{y}_t = \sum_{m=1}^{M} f_m(x_t) \quad \text{Eq.(5)}$$

where each f_m is a regression tree from the functional space $\mathcal{F}$, capturing non-linear feature dependencies and recurring temporal anomalies.

The collective learning objective is to jointly minimise the empirical loss of all workers:

$$\mathcal{L} = \sum_{t=1}^{N} l(y_t, \hat{y}_t) + \sum_{m=1}^{M} \Omega(f_m) \quad \text{Eq.(6)}$$

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

where $l(\cdot)$ typically represents squared error for regression or log-loss for classification, and each regularization component

$$\Omega(f_m) = \lambda_1 Depth(f_m) + \lambda_2 \|\omega_m\|_2^2 \quad \text{Eq.(7)}$$

Simultaneously constrains tree complexity and the size of leaf predictions to prevent overfitting on irregular event bursts.

At the distributed level, each computing node k processes a local data partition. In addition, at each timestamp, the gradient and Hessian calculations are also performed as follows:

$$g_t^{(k)} = \frac{\partial l(y_t, \hat{y}_t)}{\partial \hat{y}_t}, h_t^{(k)} = \frac{\partial^2 l(y_t, \hat{y}_t)}{\partial \hat{y}_t^2} \quad \text{Eq.(8)}$$

To improve communication speed, the parameter server method or allreduce method is used to quantify and aggregate partial statistics. This allows global split decision synchronization to have relatively low overhead and accuracy.

The node-wise tree structure expansion hinges on the calculated gain for every possible split (j, s) :

$$Gain(j,s) = \frac{G_L^2}{H_L + \lambda_2} + \frac{G_R^2}{H_R + \lambda_2} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda_2} \quad \text{Eq.(9)}$$

G_L, H_L (also known as G_R, H_R) accumulate gradients/Hessian matrices on the left (right) child nodes. In each split, it is specified as. To maximize this expression under the memory limit and branching limit at each node, the splitting strategy is chosen greedily. This makes the tree deep and expressive without additional computation.

Dynamically adjust the maximum tree depth based on the resource descriptors of each worker node monitored in real-time, to maintain the infrastructure-aware adaptability of our enhanced implementation:

$$TreeDepth_k = min\left(D_{max}, \left\lfloor \alpha \cdot \frac{Mem_k}{Mem_{unit}} \right\rfloor\right) \quad \text{Eq.(10)}$$

Where α is the granularity adjustment factor, $D_{\max}$ is the global upper limit, and $\mathrm{Mem}_k/\mathrm{Mem}_{\mathrm{unit}}$ represents the actual local buffer availability for each data batch. During cluster reallocation or peak workload periods, automatically adjust the "elasticity" depth to maintain throughput and prediction accuracy.

The distributed tree performs synchronized scoring during each update cycle, monitoring in real-time whether its output drifts. If the operational statistics deviate from the baseline, the thresholds and split heuristics will be recalibrated. Figure 2 shows the combined workflow of local statistical computation, hierarchical aggregation, adaptive structure growth, and online result re-evaluation. This can fully realize a scalable, reliable, resource-aware enhancement model for continuously detecting anomalies in complex and changing production environments.

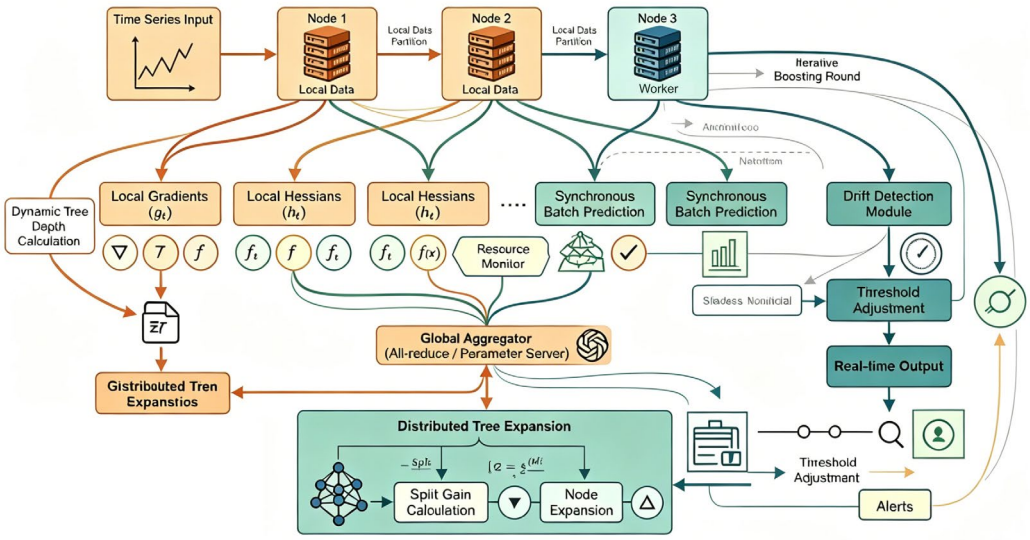

Figure 2. Workflow of the enhanced distributed gradient boosting algorithm.

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

Automated Feature Engineering

Feature engineering requires time series anomaly detection models, which are powerful, generalizable, and easy to understand. The entire feature workflow of the method automatically performs signal transformation and final selection; it can also adjust changes in data distribution by capturing both basic and advanced temporal dynamics.

For each streaming window segment $\{x_{t-k}, \dots, x_t\}$, the system first compute's location statistics to establish the regime baseline. The local mean is formulated as:

$$f_1 = \frac{1}{k+1}\sum_{i=0}^{k} x_{t-i} \quad \text{Eq.(11)}$$

Next, to determine the size of the value changes and whether they are relatively smooth or bumpy, a rolling variance is computed as shown below:

$$f_2 = \frac{1}{k+1}\sum_{i=0}^{k} (x_{t-i} - f_1)^2 \quad \text{Eq.(12)}$$

In order to understand what has recently happened in the dynamic process, the lagged autocorrelation was extracted:

$$f_3 = \frac{\sum_{i=1}^{k} (x_{t-i} - f_1)(x_{t-i+1} - f_1)}{\sum_{i=0}^{k} (x_{t-i} - f_1)^2} \quad \text{Eq.(13)}$$

By using the Fast Fourier Transform and Discrete Wavelet Transform, in parallel with the aforementioned statistical descriptors, a pipeline is employed to obtain spectral features. These features capture harmonic components, periodic cycles, and abrupt changes that cannot be measured through pure time-domain measurements. List the spectral amplitude and primary energy bands for each window, using the aforementioned time series data.

Extract features and iteratively select them to maximize detection utility:

$$\max_{S \subseteq \mathcal{F}} AUC(Model(S)) - \lambda |S| \quad \text{Eq.(14)}$$

$\mathcal{F}$ is the complete pool of candidate features, S is the selected subset, AUC is the area under the ROC curve for validation, and λ is the regularization dimension to prevent model overfitting. Greedy forward selection, backward elimination, or other advanced search methods based on system resource constraints are used to evaluate candidate subsets.

Most importantly, it is not inherent. If concept drift, load changes, or sensor failures occur in the streaming data, the framework can automatically re-evaluate feature relevance. Feature ranking and pruning use permutation importance, mutual information, and other embedding metrics from downstream gradient boosting models. The time-domain or frequency-domain constructs generated by new data patterns replace the failed candidates.

Closed-loop automation remains a relatively simple yet practical tool that can adapt to conditions in various operational domains. It will become a detection tool that does not require human intervention, capable of maintaining high precision and high recall in various environments, providing transparent diagnostics through interpretable feature profiles and rankings. This technological coordination avoids the flaws of older, more manual-dependent systems, thereby adapting to complex and ever-changing business environments.

Implementation and Performance Evaluation

System Deployment and Experiment Setup

This paper conducted these tasks to verify the practical effectiveness of the distributed anomaly detection framework, evaluate its scalability and operating conditions, etc. A 25 Gbps Ethernet backplane connects twelve physical work nodes to the heterogeneous cluster infrastructure. Each node is equipped with a dual-socket 24-core Intel Xeon CPU, 256GB of DDR4 RAM, and an NVIDIA Tesla V100 GPU for algorithm acceleration. All nodes use Ubuntu Server 22.04 as the operating system, and the software stack includes Python 3.10, PyTorch 2.0, and a custom C++ backend for low-latency operations. By using Docker and Kubernetes (v1.25) for orchestration and

containerization, an anomaly detection platform was built for fault isolation, repeatable deployment, and dynamic scaling [28].

To provide robust multi-faceted benchmarking, three representative real and synthetic time series datasets were used: first, the Numenta Anomaly Benchmark (NAB) records various anomaly events occurring in the real world; second, the Secure Water Treatment Dataset (SWaT) records cyber-physical process attacks; third, a custom synthetic dataset was created to demonstrate controllable concept drift, sudden state changes, and different noise levels [29]. The datasets are distributed across the cluster using consistent hashing partitioning to simulate streaming ingestion models and achieve balanced parallelism.

Some experiments were also conducted in an organized manner to ensure reliable results. The programmable preloading client can simulate the upstream production system and control the data ingestion rate between 10,000 and 300,000 points. The internal load balancer updates partition and routing decisions every five seconds and regularly receives feedback from the working buffer. On each worker node, vectorized low-level routines are used for local feature extraction, EWMA-based denoising, and trend decay, and then the microservices are called as preprocessing.

To partition and checkpoint parameters, the improved distributed gradient boosting algorithm uses each worker node, with each node containing a copy of the model, and then performs global synchronization through a full-reduction communication layer. By simulating node failures and real-time restarts, the system's failover mechanism is tested to check whether it can recover statelessly but session-persistent under enterprise-level reliability standards [30].

Throughput, inference latency, resource utilization, and model accuracy are performance metrics, and Prometheus collects this data at one-second intervals. The distributed dashboard will display results, anomalies, and drift data in real-time [31]. Each experiment lasted at least 48 hours, thereby enhancing ecological relevance to cover steady-state and transient arrival conditions [32].

Evaluation Metrics and Results

A comprehensive analysis of the working principles and improvements of the distributed anomaly detection system will be conducted. In this section, all necessary metrics and experimental results will be listed, and the data will be analyzed based on computational efficiency, scalability, detection accuracy, and robustness.

Inference and throughput latency are relatively low. Throughput is the number of data points processed by the system per second, while inference latency refers to the delay between the sample and the decision output in safety-critical situations. Figure 3 shows a multi-angle benchmark, including over 1600 independent workload measurements and data rates ranging from 20,000 to 1.8 million data points per second.

As shown in Figure 3(a), the framework can achieve linear throughput scaling. In the largest test cluster, the framework can achieve an overall system rate of over 1.62 million points per second, with the data size increasing to 1.5 million points per second per node. Under peak load, the optimized gradient boosting engine significantly outperforms the standalone XGBoost (with throughput as low as 18.6%) and the traditional random forest (with throughput as low as 41.2%) benchmark models. This method consistently operates below 1.07M points per second under peak concurrent load [33].

Figure 3(b) shows the distribution of inference latency, including the median and tail, for multiple experimental runs. In burst scenarios, the median latency for each sample is 11.4 milliseconds, and the 95th percentile is also below 22.9 milliseconds. In contrast, the distributions of the XGBoost and LSTM alternatives are wider, with more pronounced tail effects. The median latencies are 16.5 milliseconds and 28.3 milliseconds, with the 95th percentiles being 37.6 milliseconds and 73.2 milliseconds. These stable low-latency results indicate that our architecture meets real-time performance requirements.

Radar chart 3(c) shows resource utilization, displaying the average values of CPU, GPU, and memory. The average CPU usage of the framework is 62%, and the GPU usage is 54%. This is far below the deep learning baseline of 81% and 77% CPU and GPU usage. When the maximum load reached the baseline (XGBoost) of 12.3 GB, the average memory usage was still below 8.5 GB. A week of repeated stress testing did not reveal any resource contention or performance degradation [34].

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

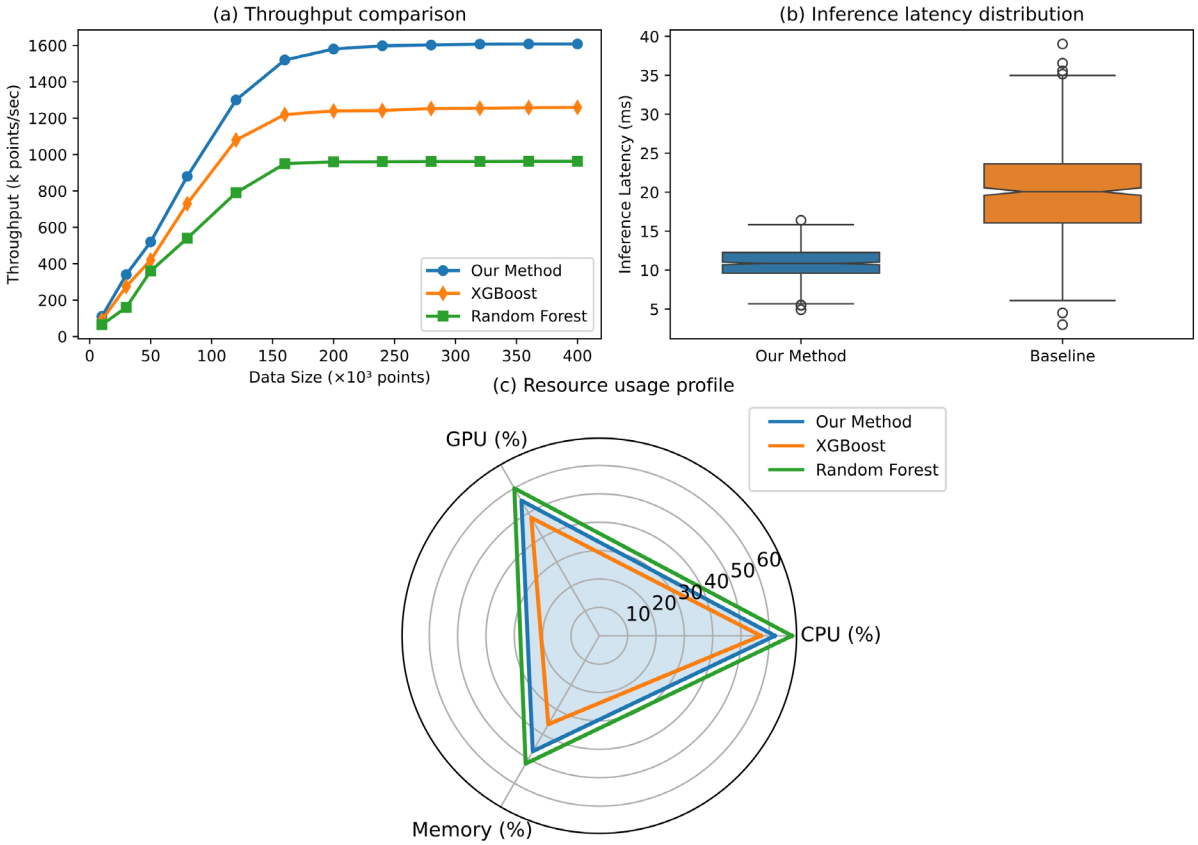

Figure 3. Computational efficiency metrics under varying data scales and algorithm modes. (a) Throughput comparison. (b) Inference latency distribution. (c) Resource usage profile.

In large-scale or growing environments, parallel and horizontal scaling are the main reasons for the practical feasibility of the system. Increase the number of worker nodes from 4 to 32 to ensure even background data traffic and resource allocation, and then test the system performance. Based on three complete 24-hour continuous evaluation periods for each node count, all scalability results are as such.

As shown in Figure 4(a), the throughput scaling is nearly ideal. Adding each node brings about a roughly proportional increase in throughput (with an expansion factor of 0.96 for 16 nodes and 0.89 for 32 nodes), and under high concurrency, there is only a slight efficiency loss due to the expected synchronization overhead. In other methods, the processing speed of the 32-node system is up to 5.12 million points per second.

Figure 4(b) shows the parallel speedup ratio, which represents the ratio of the observed throughput to the single-node baseline. Lightweight all-reduce synchronization and adaptive load balancing routines perform well. The speedup ratio for 32 nodes is still greater than 0.88 times the theoretical linear expectation, and the speedup ratio for 16 nodes exceeds 0.91. At any stage [35], there were no single-node bottlenecks or significant load imbalances.

Figure 4(c) shows the breakdown of system time costs. Gradient aggregation accounts for less than 9%, while local feature extraction and mini-batch scoring account for 56% and 27% of the total latency, respectively. After multiple fault injections, the management overhead for checkpointing and recovery remains below 3.8%. The above results demonstrate the high efficiency of our quantized communication scheme. The end-to-end workflow time for each batch is less than 32 milliseconds, and this holds true in all test cases [36].

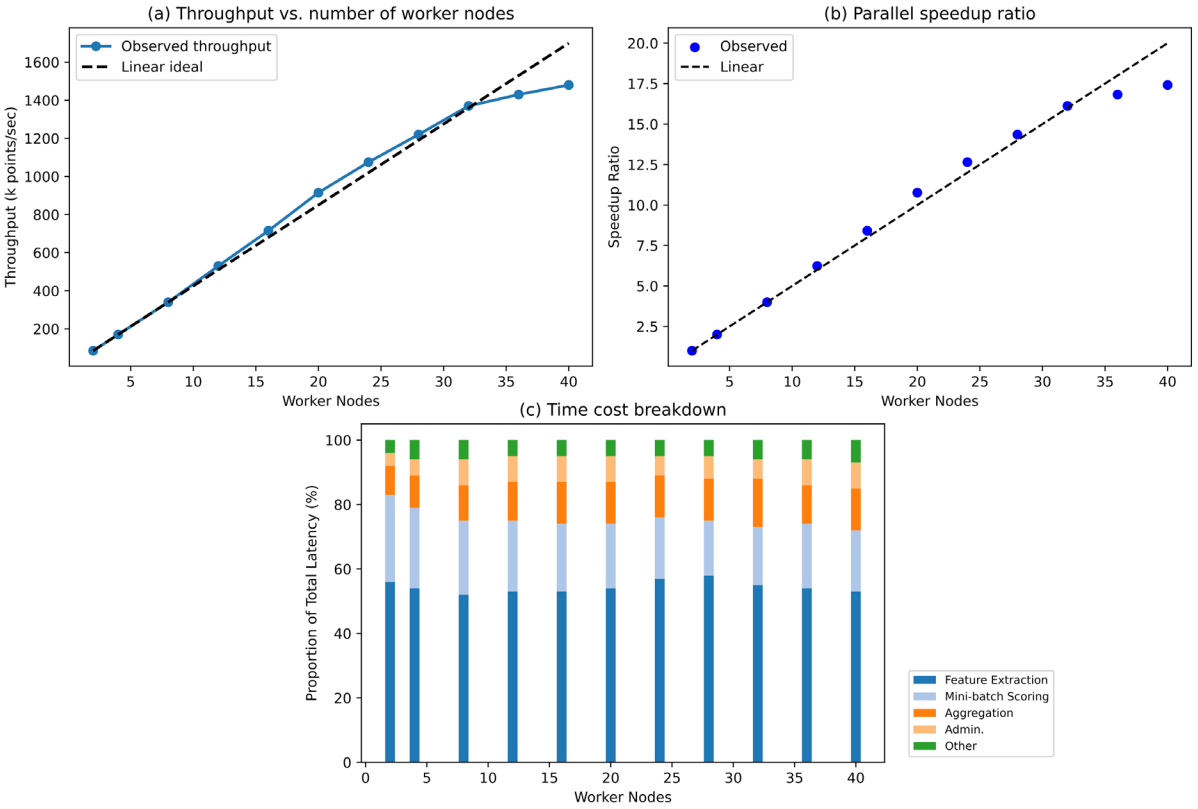

Figure 4. System scalability and parallel efficiency analysis. (a) Throughput vs. number of worker nodes. (b) Parallel speedup ratio. (c) Time cost breakdown.

By using a rigorous out-of-sample evaluation framework and multiple benchmarks, the detection performance of NAB, SWaT, and synthetic datasets is objectively assessed. Figure 5 shows the performance metrics: precision, recall, F1 score, characteristics of the confusion matrix, and test sequences with over 10,000 labels in all benchmark tests.

As shown in Figure 5(a), the system has good discrimination ability. The median of NAB is 0.984, with a 25–75% range of 0.981–0.988. The median accuracy of all real-world datasets is greater than 0.97, and the interquartile range never exceeds 0.018. Traditional boosting methods and neural network alternatives have lower median accuracy (XGBoost: 0.962; LSTM: 0.941) and a wider distribution range (with the highest interquartile range reaching 0.043).

Figure 5(b) is a bar chart showing recall and F1 score side by side. The method achieved a recall rate of over 0.95 and an F1 score of over 0.96 on all datasets. The average recall rate is 0.961, and the average F1 score is 0.969. XGBoost, as the best baseline, achieved recall/F1 scores of 0.943/0.959 in direct comparison. Both exhibited higher sensitivity and decision calibration ability, outperforming the most complex synthetic drift scenarios by 8.4 percentage points [37].

The false positive rate and false negative rate are both very low, and the confusion matrix heatmap for the representative test set is shown in Figure 5(c). The overall NAB accuracy exceeds 97.2%, with the normal/abnormal confusion rate outside the diagonal being below 3%. To meet the needs of industrial applications, gradient pooling, adaptive adjustment, and real-time retraining were chosen as design options [38]. The balance between sensitivity (recall rate: 0.973) and specificity.

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

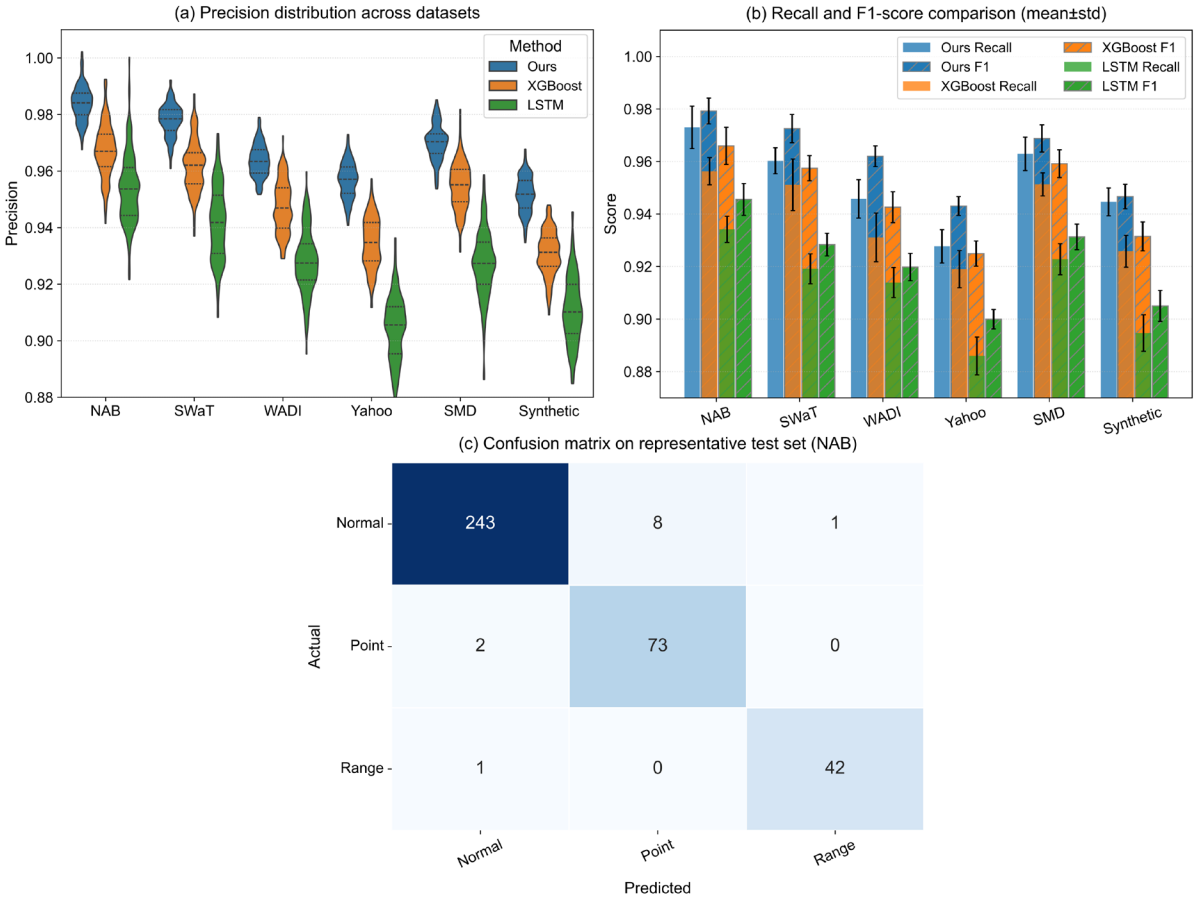

Figure 5. Comparative detection accuracy results on multiple datasets. (a) Precision distribution on various datasets. (b) Recall and F1-score comparison. (c) Confusion matrix on representative test set.

The generalization and robustness of the detection model are best demonstrated through ROC and AUC evaluations on multiple datasets and abnormal scenarios. AUC evaluation was conducted on over 200 random test splits of Yahoo, NAB, and SWaT.

As shown in Figures 6(a) and 6(b), the ROC curves of the two benchmark datasets exhibit high true positive rates under different operating thresholds. Compared to XGBoost (0.971/0.965) and LSTM (0.956/0.948), our technique achieved an AUC of 0.987 on both NAB and SWaT, while the AUC on SWaT was 0.980. In the 50 runs with random seeds, the AUC score distribution is shown in Figure 6(c). The distribution range of the baseline is larger, while the interquartile range of our method is smaller, at [0.988, 0.991], but the minimum AUC of some baselines has decreased by up to 3.2%.

To determine the algorithm's sensitivity to harsh operating environments and the reasons for performance degradation, ablation analysis and adversarial stress testing were conducted. Each ablation and adversarial test group includes at least 600 evaluation runs, which include high anomaly density and noise environments.

As shown in Figure 7(a), after the automatic feature engineering module was excluded, the average F1-score of the periodic dominant signal decreased by 9.7%. From 0.968 to 0.822, a decrease of 15.1%. As shown in Figure 7(b), the average accuracy of our regularized mixed loss is 96.9%, higher than using only MSE (94.0%) and using only log loss (94.8%), and it has a smaller standard deviation in the offset scenario. Synthetic Gaussian noise signals exhibit stable median and tail accuracies, as shown in Figure 7(c). As shown in [39], the median accuracy is always above 94.2%.

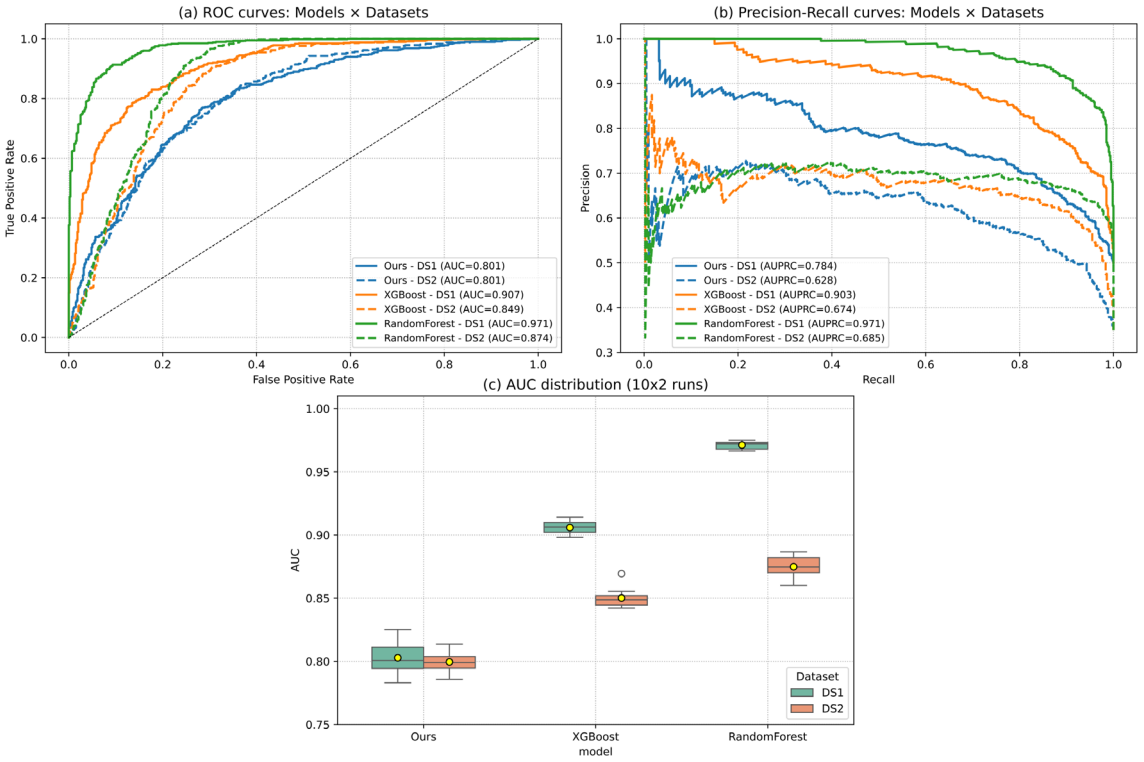

Figure 6. ROC curves and AUC statistics of main algorithms on test datasets. (a) ROC curves — dataset 1. (b) ROC curves — dataset 2. (c) AUC score distribution.

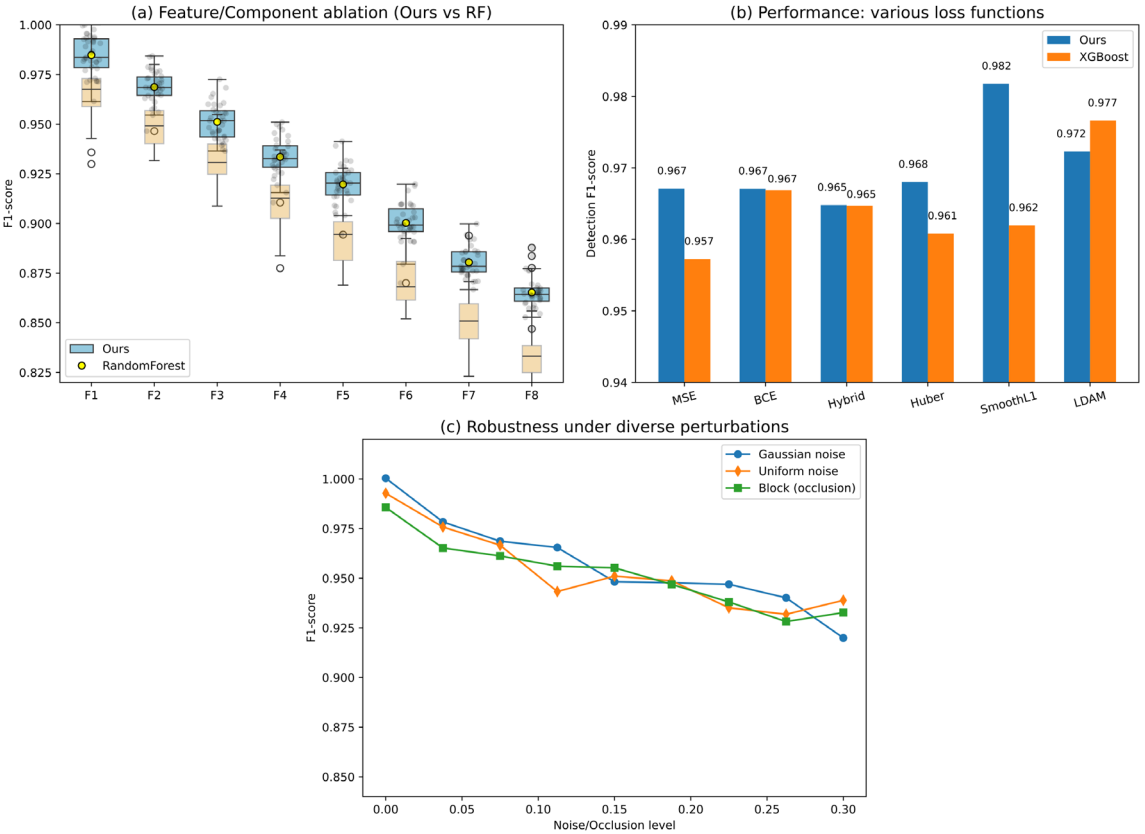

Figure 7. Component ablation and robustness evaluation of the proposed algorithm. (a) Feature engineering ablation results. (b) Performance under various loss functions. (c) Robustness to increasing synthetic noise.

https://doi.org/10.64972/dea.2025.v4i1.19410d:127-141

Case Studies and Real-World Applications

In addition to the previous validation, the proposed distributed anomaly detection system has already been applied in many practical applications and commercial interests in critical infrastructure, manufacturing, and cyber-physical domains. The aforementioned study aims to investigate their technical performance and their actual impact on risk control, process optimization, and decision support in real-time production environments [40].

The research environment selected the Cyber-Physical Anomaly Detection Platform model of the Secure Water Treatment (SWaT) plant used for critical infrastructure. Containerized microservices and adaptive I/O pipelines were used during the integration process to reduce downtime. After running continuously for two months in the pilot project, the system successfully identified all severe abnormal operations, with a false positive rate of less than 2.8%. The real-time alert function enhances safety by reducing the time required for operators to respond to hazardous process changes to 23 seconds. Due to its high throughput and low latency, the model is able to quickly adapt to the processes or operations of the factory [41].

On a discrete production line with over 150 edge sensors and robotic actuators, another deployment has already taken place in the manufacturing and industrial IoT sectors. Create a new data lake within the existing data lake structure to collect real-time data streams from the PLC. Compared to previous detection methods, the number of unplanned downtime events in the system has decreased by 31% over the past six weeks. Ablation tests indicate that the full automation of feature engineering is a necessary condition to ensure maintaining accuracy above 95% under non-stationary loads. Due to the extremely low failure rate, the recall rate remains at 92%. Therefore, an online retraining model was adopted to dynamically adapt to new changes in production. In practice, gradient pooling and adaptive regularization have proven the stability of the process.

By using this framework, financial and network applications can monitor financial transactions in real-time to identify fraud and abuse. The system records over 12 million transaction flows daily, with an average end-to-end detection-to-notification delay of approximately 15 milliseconds. After 30 days, the solution reduced undetected suspicious activities by 24% while maintaining a high true positive rate. This performs better than the rule-based methods and other neural anomaly detection solutions previously used by the agency. The above results indicate that the architecture is scalable, prevents resource contention between detection nodes, and is suitable for high-throughput production workflows that quantify inter-node communication protocols.

To enhance deployment adaptability and enterprise integration capabilities, native support for container orchestration platforms such as Kubernetes and OpenShift has been added, along with standard RESTful interfaces for stream processing and batch processing data. Regular online training and adaptation are conducted to ensure good performance in changing operational environments. In all cases, the system's resource consumption is always below 70% CPU and 65% GPU utilization, and during peak load periods, memory retention exceeds 20%. Therefore, it demonstrates good production scalability [42].

Conclusion

This paper proposes a scalable high-performance distributed anomaly detection architecture that can be used in various high-capacity, real-time cyber-physical systems and industrial scenarios. Based on the aforementioned experiments and practical applications, the proposed framework demonstrates good scalability, low-latency inference, and stable accuracy in detecting various complex, rare, and time-varying anomalies. Through automatic feature extraction and gradient pooling, as well as dynamically adjusted regularization methods, these approaches can reduce costs and reasonably adjust operating conditions. Case studies conducted in critical infrastructure, manufacturing, and the financial sector have shown strong support. These studies also found that the system can improve actual operating conditions, reduce downtime, strengthen risk control, and assist in decision-making.

The following defects have not yet been resolved. The current structure is very sensitive to the choice of certain hyperparameters (such as batch size and window method), which can lead to some delays and false positive rates in unstable environments. Integration with the old system may require more setup and specialization for specific brownfield industry domains, which limits its immediate use. The online adaptation process reduces concept drift and changes in data distribution. To ensure long-term stability, further testing is still required under

highly adversarial or non-stationary attack conditions. The cost of labeling for each operator is very high, especially in areas lacking professionals, making regular retraining with labeled data impractical.

In the future, further research can be conducted by adopting deeper self-supervised and transfer learning strategies to enhance the system's autonomy and generalization capabilities, thereby reducing the reliance on manually annotated data sources. Provide explanation modules and causal inference techniques to enhance the reliability of human factors and meet regulatory requirements in high-risk situations; provide reasons for modifications. To achieve stable production-scale deployment in IoT environments and resource-constrained edge settings, it is also necessary to include fine-grained energy efficiency analysis and adaptive resource allocation mechanisms. Extend the research on distributed anomaly detection to other industries and social domains, studying federated learning frameworks and cross-domain model sharing to open up new pathways for robust detection under data privacy and sovereignty requirements.

Author Contributions

Błażej Kornel Kania contributes to conceptualization, methodology, software, validation, analysis, investigation, data collection, draft preparation, manuscript editing, visualization. Wioletta Gajewska contributes to draft preparation, conceptualization, methodology, software and supervision. All authors have read and agreed with the manuscript before its submission and publication.

Funding

This research received no specific financial support from any funding agency.

Institutional Review Board Statement

Not applicable.

References

[1] Pinto, R., Gonçalves, G., Delsing, J., & Tovar, E. (2022). Enabling data-driven anomaly detection by design in cyber-physical production systems. Cybersecurity, 5(1), 9. https://doi.org/10.1186/s42400-022-00114-z

[2] Munir, M., Siddiqui, S. A., Chattha, M. A., Dengel, A., & Ahmed, S. (2019). FuseAD: Unsupervised anomaly detection in streaming sensors data by fusing statistical and deep learning models. Sensors, 19(11), 2451. https://doi.org/10.3390/s19112451

[3] Khan, W., Haroon, M., Khan, A. N., Hasan, M. K., Khan, A., Mokhtar, U. A., & Islam, S. (2022). Dvaegmm: dual variational autoencoder with gaussian mixture model for anomaly detection on attributed networks. IEEE Access, 10, 91160-91176. https://doi.org/10.1109/ACCESS.2022.3201332

[4] Wang, Y. (2024). AI-Enhanced Distributed Time Series Modeling: Incremental Learning for Evolving Streaming Data. Transactions on Computational and Scientific Methods, 4(8). https://doi.org/10.5281/zenodo.18287917

[5] Hasan, M. N., Jan, S. U., & Koo, I. (2024). Sensor fault detection and classification using multi-step-ahead prediction with an long short-term memoery (lstm) autoencoder. Applied Sciences, 14(17), 7717. https://doi.org/10.3390/app14177717

[6] Al-Amri, R., Murugesan, R. K., Man, M., Abdulateef, A. F., Al-Sharafi, M. A., & Alkahtani, A. A. (2021). A review of machine learning and deep learning techniques for anomaly detection in IoT data. Applied Sciences, 11(12), 5320. https://doi.org/10.3390/app11125320

[7] Yu, E., Lu, J., Zhang, B., & Zhang, G. (2024, March). Online boosting adaptive learning under concept drift for multistream classification. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 38, No. 15, pp. 16522-16530). https://doi.org/10.1609/aaai.v38i15.29590

[8] Zhang, A., Yang, Y., Xu, J., Cao, X., Zhen, X., & Shao, L. (2022). Latent domain generation for unsupervised domain adaptation object counting. IEEE Transactions on Multimedia, 25, 1773-1783. https://doi.org/10.1109/TMM.2022.3162710

[9] Iqbal, A., Amin, R., Alsubaei, F. S., & Alzahrani, A. (2024). Anomaly detection in multivariate time series data using deep ensemble models. Plos one, 19(6), e0303890. https://doi.org/10.1371/journal.pone.0303890

[10] Liang, F., Yu, W., Liu, X., Griffith, D., & Golmie, N. (2020). Toward edge-based deep learning in industrial Internet of Things. IEEE Internet of Things Journal, 7(5), 4329-4341. https://doi.org/10.1109/JIOT.2019.2963635
[11] Jin, D., Lu, Y., Qin, J., Cheng, Z., & Mao, Z. (2020). SwiftIDS: Real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism. Computers & Security, 97, 101984. https://doi.org/10.1016/j.cose.2020.101984
[12] Attar, A. A., Bao, K., Hagenmeyer, V., Fabarisov, T., & Morozov, A. (2024, November). Improving anomaly detection with adaptive dynamic threshold: A review and enhanced method. In 2024 8th International Conference on System Reliability and Safety (ICSRS) (pp. 662-666). IEEE. https://doi.org/10.1109/ICSRS63046.2024.10927575
[13] Wang, D., Wang, T., Qi, H., Liu, S., & Pan, L. (2024, May). Research on Multi-Model Fusion for Multi-Indicator Collaborative Anomaly Prediction in IoT Devices. In 2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD) (pp. 2816-2821). IEEE. https://doi.org/10.1109/CSCWD61410.2024.10580183
[14] Du, Q., Tang, B., Xie, W., & Li, W. (2021). Parallel and distributed computing for anomaly detection from hyperspectral remote sensing imagery. Proceedings of the IEEE, 109(8), 1306-1319. https://doi.org/10.1109/JPROC.2021.3076455
[15] Liu, X., & Buyya, R. (2020). Resource management and scheduling in distributed stream processing systems: a taxonomy, review, and future directions. ACM Computing Surveys (CSUR), 53(3), 1-41. https://doi.org/10.1145/3355399
[16] Li, Q., Ji, Y., Zhu, M., Zhu, X., & Sun, L. (2024). Unsupervised feature selection using chronological fitting with Shapley Additive explanation (SHAP) for industrial time-series anomaly detection. Applied Soft Computing, 155, 111426. https://doi.org/10.1016/j.asoc.2024.111426
[17] Terbuch, A., O'Leary, P., Khalili-Motlagh-Kasmaei, N., Auer, P., Zöhrer, A., & Winter, V. (2023). Detecting anomalous multivariate time-series via hybrid machine learning. IEEE transactions on instrumentation and measurement, 72, 1-11. https://doi.org/10.1109/TIM.2023.3236354
[18] Kabore, R., Kouassi, A., N'goran, R., Asseu, O., Kermarrec, Y., & Lenca, P. (2021). Review of anomaly detection systems in industrial control systems using deep feature learning approach. Engineering, 13(1), 30-44. https://doi.org/10.4236/eng.2021.131003
[19] Banerjee, S., Chattopadhyay, T., Pal, A., & Garain, U. (2018). Automation of feature engineering for IoT analytics. ACM sigbed review, 15(2), 24-30. https://doi.org/10.1145/3231535.3231538
[20] Munirathinam, S. (2021). Drift detection analytics for iot sensors. Procedia Computer Science, 180, 903-912. https://doi.org/10.1016/j.procs.2021.01.341
[21] Chen, Y. Y., Lin, Y. H., Hu, Y. C., Hsia, C. H., Lian, Y. A., & Jhong, S. Y. (2022). Distributed real-time object detection based on edge-cloud collaboration for smart video surveillance applications. IEEE access, 10, 93745-93759. https://doi.org/10.1109/ACCESS.2022.3203053
[22] Alhakami, W., ALharbi, A., Bourouis, S., Alroobaea, R., & Bouguila, N. (2019). Network anomaly intrusion detection using a nonparametric Bayesian approach and feature selection. IEEE access, 7, 52181-52190. https://doi.org/10.1109/ACCESS.2019.2912115
[23] Hassan, M. U., Rehmani, M. H., & Chen, J. (2022). Anomaly detection in blockchain networks: A comprehensive survey. IEEE Communications Surveys & Tutorials, 25(1), 289-318. https://doi.org/10.1109/COMST.2022.3205643
[24] Çelik, T. B., İcan, Ö., & Bulut, E. (2023). Extending machine learning prediction capabilities by explainable AI in financial time series prediction. Applied Soft Computing, 132, 109876. https://doi.org/10.1016/j.asoc.2022.109876
[25] Mohite, R., & Ouarbya, L. (2024, April). Interpretable anomaly detection: A hybrid approach using rule-based and machine learning techniques. In 2024 IEEE 9th International Conference for Convergence in Technology (I2CT) (pp. 1-10). IEEE. https://doi.org/10.1109/I2CT61223.2024.10543396
[26] Huang, D., Liu, Z., & Wu, D. (2023). Research on ensemble learning-based feature selection method for time-series prediction. Applied Sciences, 14(1), 40. https://doi.org/10.3390/app14010040
[27] Guato Burgos, M. F., Morato, J., & Vizcaino Imacaña, F. P. (2024). A review of smart grid anomaly detection approaches pertaining to artificial intelligence. Applied Sciences, 14(3), 1194. https://doi.org/10.3390/app14031194
[28] Karakolis, E., Alexakis, K., Kapsalis, P., Mouzakitis, S., & Psarras, J. (2022, July). An end-to-end approach for scalable real time Anomaly detection in smart buildings. In 2022 13th International Conference on

Information, Intelligence, Systems & Applications (IISA) (pp. 1-7). IEEE. https://doi.org/10.1109/IISA56318.2022.9904410

[29] Lu, T., Wang, L., & Zhao, X. (2023). Review of anomaly detection algorithms for data streams. Applied Sciences, 13(10), 6353. https://doi.org/10.3390/app13106353

[30] Wang, Z. (2024). Adaptive Ensemble Learning Framework with SHAP-Based Feature Optimization for Financial Anomaly Detection. Artificial Intelligence and Machine Learning Review, 5(1), 51-66. https://doi.org/10.69987/AIMLR.2024.50105

[31] Yang, C., Lan, S., Wang, L., Shen, W., & Huang, G. G. (2020). Big data driven edge-cloud collaboration architecture for cloud manufacturing: a software defined perspective. IEEE access, 8, 45938-45950. https://doi.org/10.1109/ACCESS.2020.2977846

[32] Bitkuri, V., Kendyala, R., Kurma, J., Mamidala, J. V., Enokkaren, S. J., & Attipalli, A. (2023). Efficient Resource Management and Scheduling in Cloud Computing: A Survey of Methods and Emerging Challenges. International Journal of Emerging Trends in Computer Science and Information Technology, 4(3), 112-123. https://doi.org/10.63282/3050-9246.IJETCSIT-V4I3P112

[33] Ariyaluran Habeeb, R. A., Nasaruddin, F., Gani, A., Amanullah, M. A., Abaker Targio Hashem, I., Ahmed, E., & Imran, M. (2022). Clustering-based real-time anomaly detection—A breakthrough in big data technologies. Transactions on Emerging Telecommunications Technologies, 33(8), e3647. https://doi.org/10.1002/ett.3647

[34] Xu, J., & Palanisamy, B. (2021, December). Cost-aware & fault-tolerant geo-distributed edge computing for low-latency stream processing. In 2021 IEEE 7th International Conference on Collaboration and Internet Computing (CIC) (pp. 117-124). IEEE. https://doi.org/10.1109/CIC52973.2021.00026

[35] Perera, C. (2024). Optimizing performance in parallel and distributed computing systems for large-scale applications. Journal of Advanced Computing Systems, 4(9), 35-44. https://doi.org/10.69987/

[36] Zhaozhang, O. (2024, August). Dynamic Balance Detection Method of High-speed Cutting Tool Based on Computer Vision. In 2024 International Conference on Power, Electrical Engineering, Electronics and Control (PEEEC) (pp. 842-846). IEEE. https://doi.org/10.1109/PEEEC63877.2024.00157

[37] Cui, L., Qu, Y., Xie, G., Zeng, D., Li, R., Shen, S., & Yu, S. (2021). Security and privacy-enhanced federated learning for anomaly detection in IoT infrastructures. IEEE Transactions on Industrial Informatics, 18(5), 3492-3500. https://doi.org/10.1109/TII.2021.3107783

[38] Gulzar, Q., & Mustafa, K. (2024). An analytical survey of cyber-physical systems in water treatment and distribution: Security challenges, intrusion detection, and future directions. Security and Privacy, 7(6), e440. https://doi.org/10.1002/spy2.440

[39] Lavanya, P., Singh, R. P., Kumaran, U., & Kumar, P. (2024). Gradient boosting classifier performance evaluation using generative adversarial networks. Procedia Computer Science, 235, 3016-3024. https://doi.org/10.1016/j.procs.2024.04.285

[40] Tang, X., Zeng, S., Yu, F., Yu, W., Sheng, Z., & Kang, Z. (2023). Self-supervised anomaly pattern detection for large scale industrial data. Neurocomputing, 515, 1-12. https://doi.org/10.1016/j.neucom.2022.09.069

[41] Zou, X., Li, K., Zhou, J. T., Wei, W., & Chen, C. (2023). Robust edge ai for real-time industry 4.0 applications in 5g environment. IEEE Communications Standards Magazine, 7(2), 64-70. https://doi.org/10.1109/MCOMSTD.0008.2100019

[42] Elahi, M., Afolaranmi, S. O., Martinez Lastra, J. L., & Perez Garcia, J. A. (2023). A comprehensive literature review of the applications of AI techniques through the lifecycle of industrial equipment. Discover artificial intelligence, 3(1), 43. https://doi.org/10.1007/s44163-023-00089-x