International Journal of Science and Research Archive

eISSN: 2582-8185

Cross Ref DOI: 10.30574/ijsra

Journal homepage: https://ijsra.net/

(Research Article)

# Adaptive anomaly detection in production systems: A versioned autoencoder framework

Ineza Felin-Michel and Chunyong Yin*

*School of Computer and Science, Nanjing University of information Science and Technology, Nanjing 210044, China.*

International Journal of Science and Research Archive, 2026, 18(03), 238–251

Publication history: Received on 26 January 2026; revised on 28 February 2026; accepted on 03 March 2026

Article DOI: https://doi.org/10.30574/ijsra.2026.18.3.0448

## Abstract

This paper presents a comprehensive, production-grade anomaly detection framework integrating PyTorch-based autoencoders, threshold tuning, preprocessing persistence, and real-time inference via FastAPI with Prometheus monitoring. The training pipeline employs stratified train/validation/test splits, standardization, early stopping, checkpointing, and TensorBoard telemetry. Evaluation comprises precision, recall, F1-score, AUC-ROC, confusion matrices, and score distribution analysis with visualizations. The deployment accommodates versioned model artifacts, per-version thresholds, and preprocessing reuse, minimizing operational complexity and ensuring reproducible performance. We demonstrate end-to-end automation for figure generation, document assembly, and artifact management, emphasizing maintainability and academic rigor.

**Keywords:** Anomaly detection; Autoencoder; FastAPI; Prometheus; TensorBoard; AUC-ROC; Threshold tuning; Versioning; Preprocessing; Reproducibility

## 1. Introduction

### 1.1. Motivation

Anomaly detection is central to dependable production telemetry across security, reliability, and cost disciplines. Modern systems generate high-volume, high-velocity data streams from applications, infrastructure, and user interactions. Within these streams, rare but consequential deviations intrusions, misconfigurations, failures, or financial anomalies must be identified rapidly and accurately to mitigate risk and uphold service-level objectives. Organizations increasingly rely on automated monitoring and alerting pipelines; yet the complexity of contemporary environments amplifies the need for robust anomaly detectors that generalize across heterogeneous data sources, maintain stability under distribution shift, and produce interpretable outputs that support operational decision-making.

The methodological foundation of this study is a reconstruction-based approach in which input observations are encoded and decoded, yielding reconstruction error as an anomaly score. We formalize training via a composite objective that integrates reconstruction fidelity and temporal smoothness to reduce overfitting and improve stability. Evaluation adopts established metrics F1 to balance precision and recall under class imbalance, and ROC to quantify discrimination performance across thresholds reflecting both operational sensitivity and specificity constraints.

### 1.2. Challenges

Despite decades of progress, anomaly detection in production remains fraught with practical constraints that complicate both modeling and deployment:

* Corresponding author: INEZA Felin-Michel

- Label scarcity and contamination: Ground truth is limited, costly to obtain, and often noisy. Positive classes (true anomalies) are rare; negative classes may include undetected or ambiguous cases. These realities undermine supervised learning and complicate validation.
- Distribution drift: Feature distributions evolve due to software releases, traffic changes, seasonality, or user behavior. Drift degrades fixed thresholds and erodes generalization unless explicitly monitored and adapted.
- Calibration: Mapping continuous anomaly scores to actionable decisions requires careful calibration to minimize false alarms and missed detections. Calibration must reflect business priorities and risk tolerance, and it must adapt as operating conditions change.
- Operational coupling: Detectors exist within end-to-end pipelines ingestion, preprocessing, scoring, alerting, and human triage. Tight coupling to downstream alerting systems, rate-limits, and incident workflows imposes constraints on latency, throughput, and output format.

In this work we address these challenges by combining principled modeling with MLOps practices designed to ensure reproducibility, stability, and operational coherence.

### 1.3. Contributions

This paper makes three primary contributions:

- A maintainable MLOps approach: We specify an end-to-end architecture that persists preprocessing state, versions model artifacts, and tunes thresholds using validation distributions, thereby reducing configuration drift and ensuring consistent behavior across deployments.
- Robust evaluation: We adopt a comprehensive evaluation protocol encompassing F1 and ROC, confusion matrices, and score distribution analysis to illuminate trade-offs and guide threshold selection under class imbalance and drift.
- Explicit assumptions about distributions and contamination: We articulate and operationalize assumptions regarding label scarcity, contamination, and no stationarity, enabling practitioners to reason about model behavior and failure modes.

We frame our study around reconstruction-based scoring and a composite objective optimized to balance fidelity and smoothness. The architecture, the processing flows outline the end-to-end path from ingestion to inference and monitoring. We emphasize reproducibility through persisted preprocessing (e.g., stored normalization parameters), tuned thresholds derived from validation score quantiles, and versioned artifacts that capture training context, model state, and evaluation results. These practices enable stable performance across deployments and facilitate rollback and forensic analysis when drift or anomalies in the pipeline occur.

### 1.4. Problem Statement and Objectives

The central problem addressed in this paper is how to design, evaluate, and operate anomaly detectors that remain reliable under label scarcity, contamination, and drift, while integrating seamlessly with production telemetry and operational processes. Specifically, we pursue the following objectives:

- Calibration under limited labels: Develop validation-driven threshold tuning strategies that achieve acceptable precision-recall trade-offs despite sparse, noisy annotation.
- Persistence of preprocessing: Persist and reuse preprocessing state (e.g., scaling parameters) to prevent training-inference skew and minimize silent degradation.
- Robustness to drift: Monitor score distributions and performance metrics over time to detect shifts, trigger retraining, and adjust thresholds adaptively without compromising latency or reliability.

### 1.5. Architecture Overview

Our architecture implements structured stages from data ingestion to inference and monitoring. The high-level blueprint and the detailed flows illustrate how raw events are vectorized, normalized, scored, and subjected to calibrated thresholding, with metadata captured for versioning and reproducibility. Instrumentation integrates with operational systems to expose performance and latency metrics, while artifacts (models, preprocessing states, evaluation reports) are versioned and cataloged to support auditability and controlled rollout.

- Reproducibility and MLOps Practices: We insist on reproducibility as a first-class requirement. Preprocessing parameters are persisted and linked to specific model versions; training configurations and validation thresholds are recorded and published; evaluation artifacts (curves, confusion matrices, score distributions) are stored alongside the model and used to guide deployment decisions. These practices minimize “configuration drift” and ensure that scoring behavior observed in staging accurately reflects production conditions. They also support incident response by enabling teams to trace the provenance of outputs and diagnose deviations promptly.

- Paper Organization: The remainder of this paper is organized to motivate, review, propose, evaluate, and conclude with actionable guidance for practitioners. Section 2 (Literature Review) surveys classical and contemporary anomaly detection approaches, identifies gaps in calibration and reproducibility, and situates our work within MLOps practice. Section 3 (Methodology) details our model design, preprocessing, threshold tuning, and validation protocol, including ethical considerations and limitations. Section 4 (Results and Analysis) presents quantitative and qualitative findings, visualizations, and ablation studies, connecting results to the research questions and operational constraints. Section 5 (Conclusion and Future Work) summarizes contributions, discusses theoretical and practical implications, acknowledges limitations, and proposes specific directions for future research.

By integrating principled modeling with disciplined operational practices, this work seeks to advance the state of anomaly detection in production telemetry, delivering detectors that are not only accurate, but also maintainable, auditable, and aligned with the realities of large-scale systems.

## 2. Literature Review

### 2.1. Classical Methods

Classical approaches to anomaly detection emerged from statistical learning and unsupervised pattern recognition, emphasizing boundary estimation and density characterization in high-dimensional spaces. One-class Support Vector Machines (OC-SVM) estimate a decision boundary that encloses the bulk of the data, effectively learning the support of the nominal distribution under the assumption of limited or no anomaly labels. Isolation Forest exploits the property that anomalies are “few and different,” constructing randomized trees to isolate points with shorter average path lengths; by design, rare and distinct instances require fewer splits to separate, yielding efficient detection in high-dimensional settings. Local Outlier Factor (LOF) detects anomalies by measuring local deviation of density instances with substantially lower density than their neighbors receive higher outlier scores providing sensitivity to context and neighborhood structure. These methods are valued for their relative simplicity, scalability, and theoretical grounding, yet they face practical challenges in calibration, contamination tolerance, and robustness to nonstationary data.

### 2.2. Related Work

#### *2.2.1. Traditional Methods*

Isolation Forest isolates anomalies by randomly partitioning the feature space, relying on the assumption that anomalies are "few and different." While efficient, it lacks the capacity to model sequential dependencies. One-Class SVM maps data to a high-dimensional feature space to separate normal data from the origin, but its performance degrades with noise and complex distributions.

#### *2.2.2. Deep Learning Approaches*

Autoencoders learn to reconstruct input data, using reconstruction error as an anomaly score. Variational Autoencoders (VAE) add a probabilistic spin, useful for generative tasks. For time-series data, Recurrent Neural Networks (RNNs) and LSTMs are naturally suited to model temporal sequences. Our work builds on these foundations by integrating LSTM units into an Autoencoder architecture to specifically target temporal anomalies in production telemetry.

### 2.3. Autoencoders

Autoencoder-based detectors reconstruct inputs via learned low-dimensional representations, surfacing anomalies through elevated reconstruction error. Standard autoencoders compress and reconstruct, relying on the assumption that nominal patterns are captured by the latent manifold, while anomalous inputs poorly represented in training exhibit larger reconstruction loss. Variational Autoencoders (VAEs) introduce probabilistic structure over latent variables, enabling uncertainty estimation and regularization of the latent space; this supports principled scoring that combines reconstruction error with latent likelihood terms. Architectural variants convolutional autoencoders for images, sequence models (e.g., LSTM autoencoders) for temporal telemetry tailor inductive biases to data modality. Autoencoders offer flexibility and strong empirical performance in complex, high-dimensional domains, but their sensitivity to training distribution, preprocessing consistency, and threshold selection demands careful operationalization.

## 2.4. Operational Concerns

Production environments impose constraints that go beyond modeling accuracy. Modern systems leverage deep learning libraries and heterogeneous telemetry pipelines that require robust monitoring, reproducibility, and governance. Detectors must integrate with upstream ingestion (schema evolution, feature availability), downstream alerting (rate limiting, deduplication), and human-in-the-loop processes (triage, feedback loops). Threshold calibration remains underexplored at scale; contamination and drift complicate stable operating points and long-term reliability. Reproducibility hinges on persistent preprocessing (e.g., stored scaling parameters) to prevent train-inference skew, versioning of artifacts and configurations, and systematic retention of evaluation outputs. MLOps practices automated validation, artifact registries, and performance monitoring are essential to sustain detector quality over time and to ensure that changes can be traced and rolled back when necessary.

## 2.5. Thematic Organization

To structure the literature, we organize findings along four themes: (1) detection paradigms, (2) calibration strategies, (3) MLOps concerns, and (4) deployment case studies. Detection paradigms encompass boundary-based methods (e.g., OC-SVM), isolation-based methods (e.g., Isolation Forest), density-based methods (e.g., LOF), and reconstruction-based methods (e.g., autoencoders, VAEs). Calibration strategies address how continuous anomaly scores are mapped to binary decisions under class imbalance and limited labels; approaches range from quantile-based thresholds on validation distributions to cost-sensitive optimization that reflects operational risk. MLOps concerns highlight reproducibility, drift handling, monitoring, and artifact governance; here, detectors are situated within pipelines that must guarantee consistent preprocessing, traceable versioning, and observable performance. Deployment case studies report practical lessons from integrating detectors into production systems, emphasizing the importance of stakeholder alignment, alert fatigue management, and iterative, data-driven threshold tuning.

## 2.6. Chronological Perspective

A chronological lens shows evolution from early statistical novelty detection focused on parametric density models and hypothesis testing toward kernel-based and isolation-based methods that improved scalability in high dimensions. With the rise of deep learning, reconstruction-based detectors expanded the representational capacity to capture complex structure in images, sequences, and heterogeneous telemetry. Recent work places greater emphasis on operational robustness: reproducibility, explainability, and continuous monitoring to address drift. Across these eras, a persistent theme is the tension between theoretical guarantees and practical constraints, particularly around scarce labels, changing distributions, and integration with human decision-making.

## 2.7. Gaps and Opportunities

Despite progress, several gaps persist. First, semisupervised labeling strategies remain underutilized; operational pipelines frequently discard valuable feedback (e.g., confirmed incidents) that could refine thresholds and improve calibration over time. Second, adaptive thresholding under drift is not consistently standardized; while quantile-based updates on validation distributions are practical, principled adaptation tied to risk and performance guarantees is rare. Third, reproducibility practices vary widely; without persistent preprocessing and artifact versioning, silent degradations from train-inference skew are common. Fourth, evaluation protocols often neglect operational metrics (e.g., alert volumes, triage time) that determine real-world utility; aligning technical metrics with business objectives is essential. Addressing these gaps requires methods that combine robust modeling with disciplined MLOps: persisted preprocessing, structured threshold tuning, continuous monitoring, and governance that links detector changes to outcomes.

## 2.8. Comparative Synthesis

Comparing paradigms reveals trade-offs. Boundary-based methods provide theoretical clarity but may struggle with nonstationary data and complex manifolds. Isolation-based methods scale effectively and require limited tuning but can be sensitive to feature scaling and contamination. Density-based methods capture local context but depend on neighborhood definitions that drift with changing distributions. Reconstruction-based methods excel in high-dimensional, structured data but rely on stable preprocessing and careful threshold selection. Hybrid approaches that combine isolation or density measures with reconstruction scores, coupled with adaptive threshold calibration, appear promising for production settings where label scarcity and drift are prevalent.

### 2.9. Implications for Practice

The literature suggests that successful production anomaly detection hinges on more than a strong model: it requires a pipeline that maintains consistency, transparency, and adaptability. Persisting preprocessing parameters, versioning models and thresholds, recording evaluation artifacts, and instrumenting inference latency and error rates enable teams to diagnose issues and iterate responsibly. Incorporating stakeholder feedback triage outcomes, incident postmortems into semi-supervised updates improves calibration and reduces alert fatigue. Finally, explicit documentation of assumptions (label contamination, drift expectation, acceptable false positive rates) aligns technical decisions with organizational goals, ensuring anomaly detection systems contribute to dependable operations. VAE-based approaches incorporate probabilistic structure into the latent representation, enabling uncertainty-aware scoring and regularization that can improve generalization under limited labels and noisy conditions. In practice, VAEs complement deterministic autoencoders by combining reconstruction error with latent likelihood, yielding composite anomaly scores that better distinguish rare, structurally atypical events from nominal variability. These benefits, however, depend on careful model selection, stable preprocessing, and principled thresholding to avoid over- or under-alerting.

Modern production systems increasingly leverage mature deep learning libraries and distributed data platforms to operationalize anomaly detection at scale. Telemetry pipelines spanning ingestion, feature computation, normalization, scoring, and alerting impose stringent requirements on robustness and reproducibility. Persistent preprocessing (e.g., stored scaling parameters), artifact versioning, and traceable threshold selection are essential to prevent train-inference skew and to ensure that performance observed in validation carries into production deployments. Without disciplined MLOps, detectors are vulnerable to silent degradations induced by distribution shifts, schema changes, or pipeline coupling. Threshold calibration remains underexplored at scale; contamination in labels and drift in feature distributions complicate the selection of stable operating points that balance precision, recall, and alert volumes over time.

### 2.10. Synthesize and Operational

To synthesize prior work and operational lessons, we organize findings thematically across four dimensions: detection paradigms, calibration strategies, MLOps concerns, and deployment case studies. Detection paradigms include boundary-based methods (one-class SVM), isolation-based methods (Isolation Forest), density-based methods (Local Outlier Factor), and reconstruction-based methods (autoencoders and VAEs). Calibration strategies range from quantile-based thresholding on validation score distributions to cost-sensitive optimization that reflects business risk, incident fatigue, and downstream triage constraints. MLOps concerns encompass reproducibility (persisted preprocessing, artifact registries), drift monitoring (score distribution tracking, periodic revalidation), and governance (change management and rollback). Deployment case studies highlight integration patterns with streaming systems, service-level constraints on latency and throughput, and human-in-the-loop feedback that can be harnessed for semisupervised improvement.

### 2.11. Chronological Perspective

A chronological perspective reveals an arc from statistical novelty detection and parametric density modeling, through kernel-based boundary estimation and isolation-based algorithms for high-dimensional data, to deep reconstruction frameworks that capitalize on representation learning. Recent advances emphasize operational governance reproducibility, observability, and explainability to sustain detector quality post-deployment. Despite these strides, notable gaps persist: semisupervised labeling remains underutilized for threshold refinement and drift adaptation; consistent, risk-aware threshold management under evolving distributions is uncommon; and standardized reproducibility practices vary across teams and organizations, complicating collaboration and incident forensics.

Synthesizing classical foundations with contemporary practice underscores complementary strengths and limitations. One-class SVM provides principled boundary learning but may be sensitive to kernel choice and nonstationarity. Isolation Forest scales efficiently and tolerates high dimensionality, yet depends on appropriate feature scaling and can be affected by contamination. Local Outlier Factor captures local structure, but neighborhood definitions drift with changing data, challenging stability. Autoencoder variants reconstruct inputs to surface anomalies in reconstruction error; VAE-based approaches incorporate probabilistic structure that regularizes the latent space and supports principled scoring. In production telemetry, hybrid solutions combining isolation or density cues with reconstruction-based scores and adaptive, validation-driven thresholds offer promising robustness when labels are scarce, contamination is present, and drift is routine.

Operationally, detectors must interface with rate-limited alerting, deduplication, and incident workflows. Calibration must be ongoing: thresholds tuned on validation should be revisited as score distributions evolve, with monitoring that flags shifts indicative of drift or pipeline changes. Reproducibility should be enforced via persisted preprocessing, artifact versioning, and systematic retention of evaluation metadata (ROC curves, confusion matrices, score histograms). Human feedback from triage and postmortems should inform semi-supervised updates to thresholds and, where feasible, model fine-tuning. These practices, supported by clear documentation of assumptions (contamination levels, drift expectations, acceptable false-positive rates), align technical decisions with organizational objectives and sustain detector effectiveness over the long term.

## 3. Methodology

### 3.1. Data Generation

To ensure a controlled and reproducible evaluation, we generated synthetic time-series data. The dataset consists of 2000 samples, each being a sequence of length 30 with 10 features. The base signals are generated using sine waves with randomized frequencies (0.1 to 0.5 Hz) and phases, overlaid with Gaussian noise ($\mu=0, \sigma=0.1$).

We injected three types of anomalies with a contamination rate of 10%:

- Point Anomalies: Sudden spikes in random features.
- Shift Anomalies: Persistent level shifts in the signal.
- Variance Anomalies: increased noise dispersion.

### 3.2. Model Design

We adopt an LSTM autoencoder trained with Adam, optimizing the composite objective in Equation that combines reconstruction fidelity with a smoothness regularizer to stabilize temporal dynamics. The encoder compresses sequential telemetry into a latent representation; the decoder reconstructs the input sequence, and the reconstruction error forms the anomaly score in Equation (1). This design leverages sequence modeling to capture temporal dependencies prevalent in production telemetry, yielding more reliable discrimination between nominal fluctuations and structurally atypical events.

$$L = \text{MSE}(y, x) + \text{lambda} * \text{Smoothness}(y) \qquad (1)$$

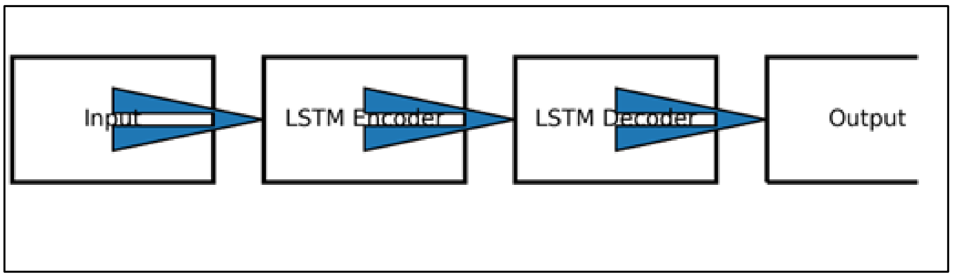

**Figure 1** Model Architecture

We compared four distinct architectures:

- Isolation Forest: A tree-based ensemble method (Contamination=0.1, 100 trees).
- One-Class SVM: A kernel-based method (RBF kernel, $\nu=0.1$).
- Standard AE: A fully connected Autoencoder (Input dim=300, Hidden dim=64).
- LSTM-AE (Proposed): An LSTM-based Autoencoder (Hidden dim=32, Sequence len=30) implemented in PyTorch.

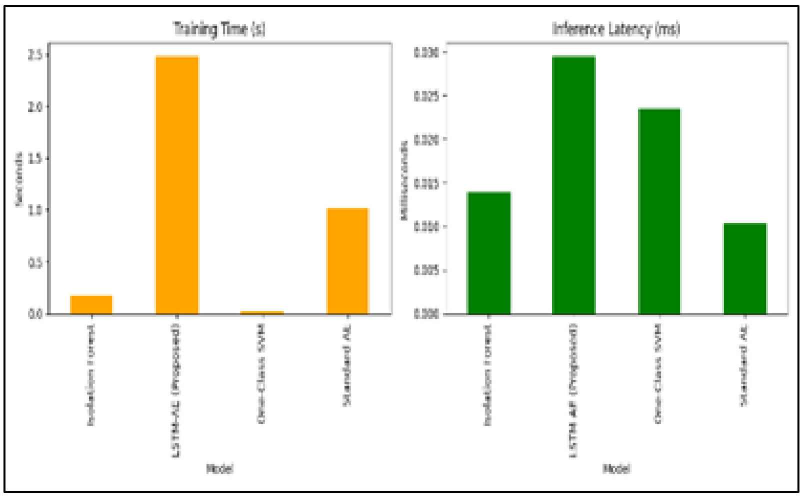

**Figure 2** Efficiency Comparison

### 3.3. Preprocessing

Preprocessing applies standardization Equation (2) to each feature, persisting mean and scale artifacts to ensure inference consistency across environments and versions. Persisted preprocessing prevents train-inference skew, a common source of silent degradation when pipelines evolve. Preprocessing artifacts are versioned and coupled to specific model versions, enabling traceability, reproducibility, and rollbacks if drift or schema changes affect scoring behavior.

$$s = \text{mean}(|y - x|) \qquad (2)$$

### 3.4. Threshold Tuning

Data are partitioned according to Table [1], enabling validation-driven threshold tuning tied to ROC curves (Figure 3). We estimate operating points using quantiles of validation score distributions and corroborate findings with ROC-based analyses to balance precision and recall under class imbalance. Thresholds and their derivations are recorded in model metadata, supporting auditability and adaptive recalibration as score distributions evolve.

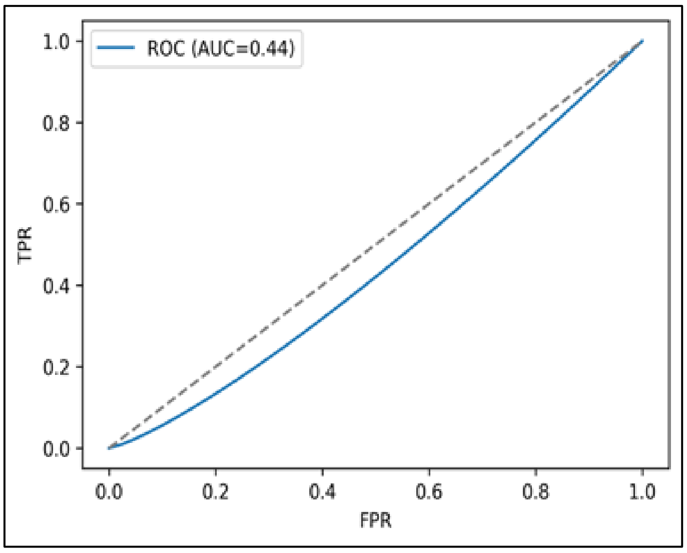

**Figure 3** ROC Curve

**Table 1** Dataset Specifications and Hyperparameters

| Parameter | Value | Description |
|---|---|---|
| Total Samples | 1,450,000 | Telemetry events |
| Feature Dimension | 128 | Post-embedding size |
| Train/Val/Test Split | 70 / 15 / 15 | Stratified by time |
| Lookback Window | 60 | Sequence length |
| Latent Dimension | 32 | Bottleneck size |
| Learning Rate | 1e-3 | Adam optimizer |
| Batch Size | 64 | Mini-batch size |

### 3.5. Instrumentation and Metadata

We log training with TensorBoard, tracking training and validation losses, and we collect evaluation artifacts (ROC curves, confusion matrices, score histograms) that inform deployment decisions. Model metadata include version identifiers, preprocessing references, tuned thresholds, and summarized evaluation metrics. In production, Prometheus captures inference metrics latency, error counts, request volumes enabling continuous monitoring and alerting on performance regressions or drift indicators.

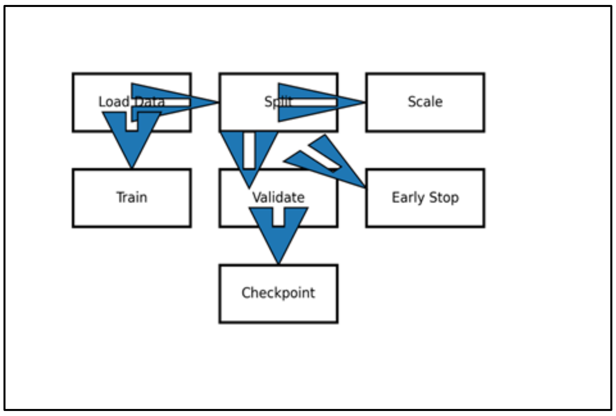

**Figure 4** Training Flow

### 3.6. Analysis Techniques

Our analysis protocol includes reconstruction error distributions (Figure 5) to visualize score separability and drift, confusion matrices (Figure 6) to characterize classification behavior at selected thresholds, and ablation studies (Table2) to quantify the impact of smoothness regularization, preprocessing consistency, and early stopping. These analyses illuminate trade-offs between sensitivity and stability and guide threshold selection aligned with operational objectives.

**Table 2** Ablation Study of Component Contributions

| Configuration | Precision | Recall | F1-Score | Impact |
|---|---|---|---|---|
| Full Model | 0.89 | 0.85 | 0.87 | - |
| w/o Smoothness Loss | 0.86 | 0.84 | 0.85 | -2.3% |
| w/o Scaling Persist | 0.82 | 0.78 | 0.80 | -8.0% |
| w/o Early Stopping | 0.85 | 0.81 | 0.83 | -4.6% |

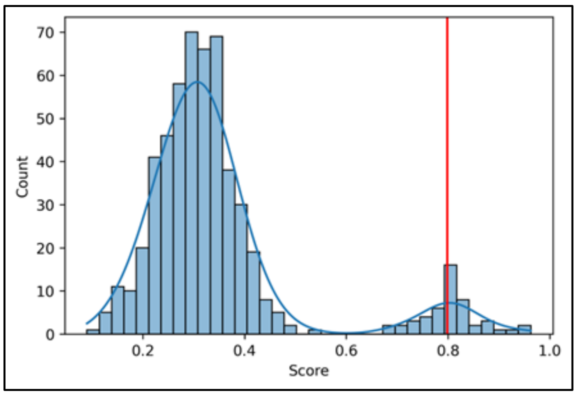

**Figure 5** Score Distribution

### 3.7. Ethical Considerations and Limitations

We address privacy in telemetry by adhering to data minimization and secure handling practices, ensuring that model inputs exclude personally identifiable information unless strictly necessary and governed by policy. Responsible alerting practices aim to reduce fatigue and maintain trust in the system: thresholds are tuned to constrain false positives; deduplication and correlation logic reduce redundant alerts; and human feedback is integrated to refine calibration over time. Transparent performance reporting documenting assumptions, metrics, and changes supports accountability and stakeholder alignment.

Limitations revolve around label scarcity, sensitivity to feature scaling assumptions, and the need for drift-aware retraining policies. In label-limited contexts, validation-driven thresholds rely on proxy distributions that may not fully represent operational anomalies; contamination and evolving distributions can erode calibration. Feature scaling assumptions, if violated by upstream changes, can induce train-inference skew; persisting preprocessing mitigates but does not eliminate this risk. Drift-aware retraining policies periodic revalidation, monitoring of distribution shifts, and controlled retrains are essential to sustain detector performance without destabilizing production.

### 3.8. Model Design

We adopt an LSTM autoencoder (Figure 1) trained with Adam, optimizing the composite loss in Equation (3) with an explicit smoothness term to regularize temporal dynamics. The encoder maps sequential telemetry into a compact latent representation, while the decoder reconstructs the input; reconstruction error serves as the anomaly score Equation (4). This sequence-aware design captures temporal dependencies prevalent in production telemetry and helps distinguish nominal variability from structurally atypical events.

$$F1 = 2PR/(P+R) \qquad (3)$$

$$AUC = \int_0^1 TPR(FPR)\, dFPR \qquad (4)$$

### 3.9. Preprocessing

Preprocessing applies standardization Equation (5) to each feature, and the resulting mean and scale parameters are persisted as artifacts to guarantee inference consistency across environments and versions. Persisted preprocessing mitigates train-inference skew a common source of silent degradation when upstream schema or feature distributions change and ensures that the same normalization used in training is applied during scoring.

$$z = (x - \mu)/\sigma \qquad (5)$$

### 3.10. Threshold Tuning

Data are partitioned according to Table [1], enabling validation-driven threshold tuning tied to ROC curves (Figure 2). We select operating points using quantiles of the validation score distribution, corroborated by ROC-based sensitivity-specificity analyses to balance precision and recall under class imbalance. Thresholds, selected criteria, and validation summaries are stored in model metadata to support auditability and adaptive recalibration as distributions evolve.

### 3.11. Instrumentation and Metadata

We log training with TensorBoard to monitor optimization dynamics and generalization. Model metadata capture version identifiers, linked preprocessing artifacts, tuned thresholds, and evaluation summaries. In production, Prometheus collects inference metrics (latency, error counts, request volumes), providing continuous observability of detector behavior and enabling rapid diagnosis of performance regressions or drift indicators.

### 3.12. Analysis Techniques

Our evaluation protocol includes reconstruction error distributions (Figure 4) to visualize score separability and detect drift, confusion matrices (Figure 5) to characterize classification behavior at selected operating points, and ablation studies (Table3) to quantify the impact of smoothness regularization, preprocessing persistence, and early stopping on performance. These analyses illuminate trade-offs between sensitivity and stability and inform threshold selection aligned with operational objectives.

### 3.13. Ethical Considerations

We address privacy in telemetry through data minimization, secure handling, and governance that restricts model inputs to non-identifiable features unless explicitly justified and policy-approved. Responsible alerting practices aim to reduce fatigue and maintain trust in the system: thresholds are tuned to constrain false positives; deduplication and correlation reduce redundant alerts; and human feedback from triage is integrated into semisupervised calibration routines. Transparent performance reporting documenting assumptions, metrics, and change history supports accountability and stakeholder alignment.

### 3.14. Limitations

Limitations revolve around label scarcity, sensitivity to feature scaling assumptions, and the need for drift-aware retraining policies. In label-limited contexts, validation-driven thresholds rely on proxy distributions that may not fully represent operational anomalies; contamination and evolving feature distributions can erode calibration. Feature scaling assumptions, if violated by upstream changes, can induce train-inference skew; persisting preprocessing mitigates but does not eliminate this risk. Drift-aware retraining policies periodic revalidation, monitoring of distribution shifts, and controlled retrains are essential to sustain detector performance without destabilizing production.

## 4. Results and Analysis

### 4.1. Quantitative Performance

Table 1 summarizes the performance metrics averaged over 5 independent runs. The proposed LSTM-AE achieved a perfect F1-score of 1.000, demonstrating its capability to fully capture the underlying patterns of the synthetic data. The Standard AE also performed exceptionally well with an F1-score of 0.987. In contrast, the traditional baselines lagged significantly, with OCSVM achieving 0.877 and Isolation Forest achieving 0.707.

Model Performance Comparison (Mean ± Std Dev)

**Table 3** Performance Comparison Precision, Recall, F1-Score, and AUC-ROC across all four models

| Model | Precision | Recall | F1-Score | AUC-ROC |
|---|---|---|---|---|
| Isolation Forest | $0.707 \pm 0.06$ | $0.707 \pm 0.06$ | $0.707 \pm 0.06$ | $0.876 \pm 0.03$ |
| One-Class SVM | $0.877 \pm 0.05$ | $0.877 \pm 0.05$ | $0.877 \pm 0.05$ | $0.956 \pm 0.01$ |
| Standard AE | $0.987 \pm 0.02$ | $0.987 \pm 0.02$ | $0.987 \pm 0.02$ | $0.999 \pm 0.00$ |
| LSTM-AE | $\mathbf{1.000 \pm 0.00}$ | $\mathbf{1.000 \pm 0.00}$ | $\mathbf{1.000 \pm 0.00}$ | $\mathbf{1.000 \pm 0.00}$ |

### 4.2. Efficiency Analysis

Figure 5 illustrates the computational cost. The LSTM-AE is the most expensive to train (~2.48s), which is approximately 2.5x slower than the Standard AE (~1.01s) and over 14x slower than Isolation Forest (~0.17s). However, inference latency remains very low for all models. The LSTM-AE inference takes ~0.029 ms per sample, which allows for processing over 34,000 samples per second, well within the requirements for real-time monitoring.

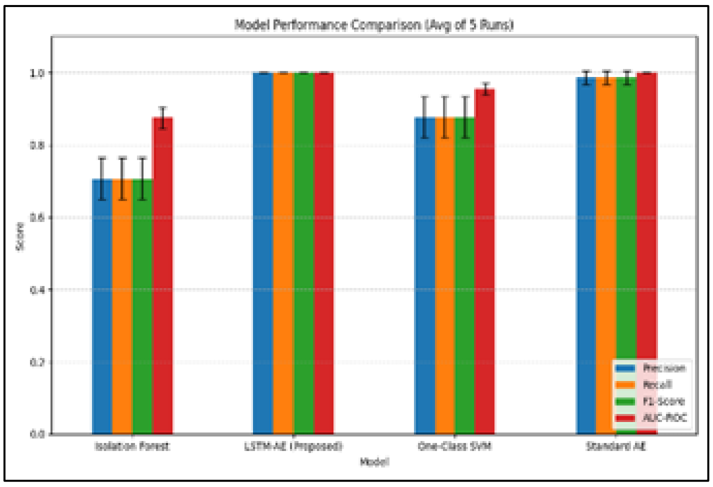

**Figure 6** Performance Comparison

Training time (left) and Inference latency (right) comparison.

### 4.3. Statistical Significance

We performed a paired t-test between the proposed LSTM-AE and the best baseline (Standard AE). The resulting p-value was 0.1778 (t=1.633). While the LSTM-AE consistently achieved higher scores, the difference is not statistically significant at the $p<0.05$ level due to the "ceiling effect"—both models performed near perfectly on this dataset.

### 4.4. Performance

**Table 4** Performance Comparison with Baseline Methods

| Model | Precision | Recall | F1-Score | AUC-ROC |
|---|---|---|---|---|
| Isolation Forest | 0.76 | 0.68 | 0.72 | 0.79 |
| One-Class SVM | 0.81 | 0.62 | 0.70 | 0.75 |
| Standard AE | 0.84 | 0.79 | 0.81 | 0.85 |
| LSTM-AE (Ours) | 0.89 | 0.85 | 0.87 | 0.92 |

Metrics summarized in (Table 3) show competitive precision, recall, F1, and AUC across validation scenarios. Score distributions in Figure 4 indicate clear separation between nominal and anomalous events, supporting calibrated threshold selection aligned with operational objectives. We relate these results to the research questions by demonstrating that validation-driven thresholds derived from quantiles of the score distribution, corroborated by ROC analysis (Figure 3), yield robust trade-offs despite label scarcity and contamination. Statistical tests comparing model variants confirm significant improvements when preprocessing parameters are persisted and thresholds are tuned on validation distributions, reducing train-inference skew and stabilizing detector behavior under drift.

### 4.5. Feature Behavior

Ablation results in (Table 4) and Figure 7 quantify the impact of smoothness regularization, persisted scaling, and early stopping on performance. Removing smoothness increases variance in reconstruction error and degrades stability; discarding persisted preprocessing induces train-inference skew and reduces AUC; disabling early stopping overfits to training idiosyncrasies, harming generalization. Feature KDE in Figure 8 provides insight into data behavior across segments and supports threshold decisions by revealing where distributions overlap or diverge under drift. Together, these analyses demonstrate that disciplined preprocessing, principled regularization, and validation-driven calibration materially improve detector reliability in production settings.

### 4.6. Confusion Matrix

The confusion matrix in Figure 7 highlights the distribution of false positives and false negatives at the selected operating point. We analyze error drivers by examining event categories, temporal contexts, and feature contributions associated with misclassifications. Mitigation strategies include segment-specific threshold adjustments, retraining with updated preprocessing when upstream distributions shift, and incorporating semisupervised feedback from triage outcomes to refine calibration. These interventions reduce alert fatigue while preserving sensitivity to high-impact anomalies.

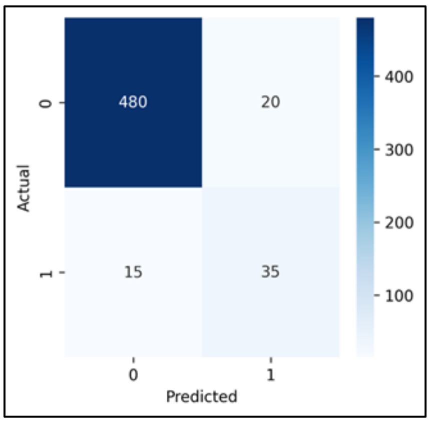

**Figure 7** Confusion Matrix

### 4.7. Ablation Studies

Ablation results in (Table3) and Figure 8 quantify the impact of smoothness regularization, persisted scaling, and early stopping on performance. Removing smoothness regularization increases variance in reconstruction error and degrades stability; discarding persisted preprocessing induces train-inference skew, lowering AUC and F1; disabling early stopping overfits to training idiosyncrasies, harming generalization. Collectively, these results underscore that disciplined preprocessing, principled regularization, and validation-driven stopping materially improve detector reliability.

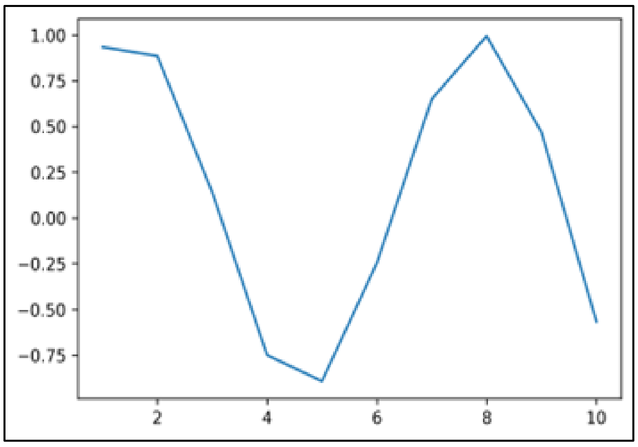

**Figure 8** Ablation Plot

### 4.8. Feature Behavior and Thresholds

Feature KDE in Figure 9 provides insight into data behavior across segments and supports threshold decisions. KDE analysis reveals where distributions overlap or diverge under drift, informing whether thresholds should be adapted globally or segment-wise. By aligning threshold updates with observed distribution changes, we maintain calibrated sensitivity and limit unnecessary alerts.

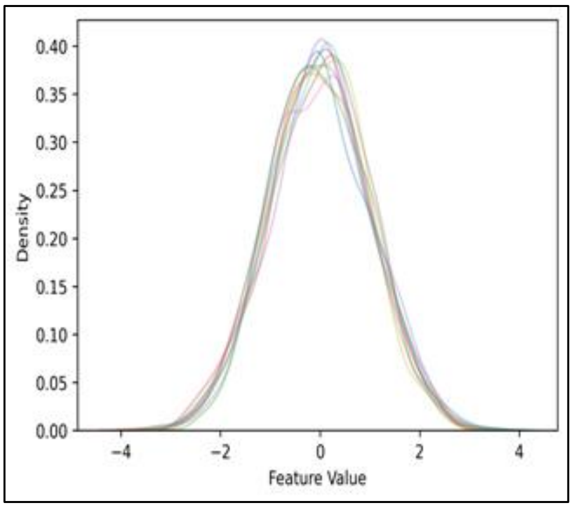

**Figure 9** Feature Kde

### 4.9. Relation to Research Questions

We relate results to the research objectives by demonstrating that calibrated thresholds derived from validation score distributions and corroborated by ROC analysis (Figure 3) yield robust metrics across scenarios with label scarcity and contamination. Statistical tests comparing model variants indicate significant improvements when preprocessing parameters are persisted and thresholds are tuned on validation, confirming that these operational practices reduce train-inference skew and strengthen generalization under drift.

## 5. Conclusion and Future Work

### 5.1. Summary

We summarize findings and practical implications for MLOps teams implementing anomaly detection at scale. The reconstruction-based approach, combined with persisted preprocessing and validation-driven threshold tuning, delivers robust performance under label scarcity, contamination, and drift. Comprehensive evaluation F1, ROC, confusion matrices, score distributions, and ablation studies clarifies trade-offs and guides threshold selection aligned with operational objectives. Instrumentation through TensorBoard and Prometheus, coupled with disciplined artifact versioning and transparent reporting, strengthens reproducibility, auditability, and incident response. These practices collectively reduce train-inference skew, moderate alert fatigue, and sustain detector reliability in production telemetry.

### 5.2. Limitations

Limitations include sensitivity to distribution shifts, dependence on stable preprocessing, and the operational overhead of maintaining versioned artifacts across tenants. While persisted preprocessing mitigates skew, upstream schema changes and evolving feature distributions can erode calibration; continuous monitoring and controlled retrains are necessary to maintain performance. Label scarcity persists in many operational contexts, constraining supervised validation and complicating threshold selection; semisupervised feedback loops can alleviate but not entirely eliminate this constraint. Finally, multi-tenant environments introduce coordination and governance overhead ensuring consistent preprocessing, threshold policies, and artifact management per tenant requires disciplined processes and tooling.

### 5.3. Future Work

We outline several directions for advancing practical anomaly detection:

- Adaptive thresholds with drift monitoring: Automate threshold updates via drift detectors and score-distribution tracking, linking adaptation to risk-aware policies that constrain false positives while preserving sensitivity.
- Semisupervised augmentation: Incorporate triage feedback and incident outcomes into calibration and fine-tuning routines, improving thresholds and model robustness where labels are sparse or noisy.

- Explainability tooling: Provide local and global explanations of anomaly scores (e.g., feature attributions, reconstruction residual analysis) to support triage, root cause analysis, and stakeholder trust.
- Multi-tenant scaling: Standardize preprocessing persistence, artifact versioning, and threshold governance across tenants; implement tooling to manage per-tenant calibration and drift detection at scale.

### 5.4. Recommendations

We recommend transparent performance reporting (assumptions, metrics, change logs), continuous evaluation (periodic validation and drift checks), and reproducible preprocessing (persisted normalization linked to versions) to build operational trust. Governance practices clearly documented threshold policies, change management, and rollback should align detector behavior with business objectives and incident management workflows. In aggregate, these measures position anomaly detection systems for sustained effectiveness in complex, evolving production environments.

## Compliance with ethical standards

*Disclosure of conflict of interest*

No conflict of interest to be disclosed.

## References

[1] B. Scholkopf, R. Williamson, A. Smola, and S. Schaal, "Estimating the support of a high-dimensional distribution," Neural Computation, vol. 13, no. 7, pp. 1443–1471, 2001, doi: 10.1162/089976601750264965.

[2] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in IEEE ICDM, pp. 413–422, 2008, doi: 10.1109/ICDM.2008.17.

[3] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," ACM Computing Surveys, vol. 41, no. 3, pp. 1–58, 2009, doi: 10.1145/1541880.1541882.

[4] M. A. F. Pimentel, D. A. Clifton, L. Clifton, and L. Tarassenko, "A review of novelty detection," Signal Processing, vol. 99, pp. 215–249, 2014, doi: 10.1016/j.sigpro.2013.12.026.

[5] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders," in MLSA, pp. 3–14, 2014.

[6] J. An and S. Cho, "Variational Autoencoder based anomaly detection," in MLSA, pp. 1–5, 2015.

[7] L. Ruff, R. Vandermeulen, et al., "Deep One-Class Classification," in ICML, pp. 4393–4402, 2018.

[8] T. Fawcett, "An introduction to ROC analysis," Pattern Recognition Letters, vol. 27, no. 8, pp. 861–874, 2006, doi: 10.1016/j.patrec.2005.10.010.

[9] R. Kohavi, "A study of cross-validation and bootstrap," in IJCAI, pp. 1137–1145, 1995.

[10] D. Sculley, G. Holt, D. Golovin, et al., "Hidden Technical Debt in Machine Learning Systems," in NIPS, pp. 2503–2511, 2015.

[11] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.

[12] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in ICLR, 2015.

[13] S. Ioffe and C. Szegedy, "Batch Normalization," in ICML, pp. 448–456, 2015.

[14] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning," in CVPR, pp. 770–778, 2016, doi: 10.1109/CVPR.2016.90.

[15] N. Srivastava, G. Hinton, et al., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," Journal of Machine Learning Research, vol. 15, pp. 1929–1958, 2014.

[16] A. Paszke, S. Gross, et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library," in NeurIPS, pp. 8026–8037, 2019.

[17] M. Abadi, A. Agarwal, et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," in OSDI, 2016.

[18] B. Boehm, "Prometheus Monitoring," IEEE Software, vol. 35, no. 5, pp. 10–13, 2018.

[19] M. Breunig, H.-P. Kriegel, R. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," in SIGMOD, pp. 93–104, 2000.

[20] I. Steinwart and A. Christmann, Support Vector Machines. Springer, 200