HIERARCHICAL CHANGE SIGNATURE ANALYSIS: A FRAMEWORK FOR ONLINE DISCRIMINATION OF INCIPIENT FAULTS AND BENIGN DRIFTS IN INDUSTRIAL TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Industrial fault detection systems often struggle to distinguish benign operational drifts (e.g., tool wear, recipe changes) from incipient faults, frequently adapting to faults as new "normal" states and risking catastrophic failures. This work proposes a hierarchical framework that decouples change detection from change characterization. When a drift is detected, the system generates a Multi-Scale Change Signature (MSCS) that quantifies geometric and statistical transformations in the primary detector's latent space. An unsupervised Drift Characterization Module (DCM), trained on an Online Normality Baseline (ONB), classifies each signature as benign or potentially faulty. Benign drifts are ignored, while potential faults are flagged for review; confirmed benign drifts are incorporated into the ONB for future adaptation. The framework is model-agnostic, computationally efficient, and scalable through a tiered human-in-the-loop mechanism. Experiments on the Tennessee Eastman Process dataset with injected drifts and faults demonstrate high fault detection rates, fewer false alarms, and efficient adaptation to benign changes.

1 INTRODUCTION

Industrial fault detection systems are critical for maintaining operational safety and efficiency in modern manufacturing processes (Ruppert et al., 2018; Ahi & Nouri, 2025). These systems must continuously monitor complex time-series data streams to identify anomalies that may indicate equipment degradation, process upsets, or safety hazards (Zhou & Li, 2024; Eivaghi & Bazin, 2024; Xu & Wang, 2025). However, industrial processes are inherently dynamic, with operational conditions constantly evolving due to factors such as equipment aging, environmental changes, production modifications, and maintenance activities (Ahi & Nouri, 2025; Ruppert et al., 2018).

A fundamental challenge in industrial fault detection lies in distinguishing between *benign operational drifts* and *incipient faults*. Benign drifts represent normal variations in process behavior that do not indicate equipment problems, such as gradual tool wear, recipe adjustments, or seasonal environmental changes. In contrast, incipient faults are early-stage equipment malfunctions that, if undetected, can lead to catastrophic failures, safety incidents, or significant production losses (Sobhani & Ghaemi, 2011; Nasif & Chen, 2024; Dissem & Brown, 2024).

Traditional fault detection systems often struggle with this discrimination, leading to two critical problems: **(1)** *False alarms* caused by misclassifying benign drifts as faults, which result in unnecessary production interruptions and operator fatigue, and **(2)** *Missed detections* when the system adapts to incipient faults, treating them as normal states and failing to alert operators to developing problems.

We propose a hierarchical framework that addresses these limitations through a three-stage approach: **(1) Change Detection** using a primary detector to flag anomalies, **(2) Change Characterization** via Multi-Scale Change Signature (MSCS) generation, and **(3) Change Classification** using an unsupervised Drift Characterization Module (DCM) against an Online Normality Baseline (ONB). This approach enables the system to make informed decisions about whether to adapt to detected changes or flag them for human review, thereby reducing both false alarms and missed detections.

Our main contributions include: **(1)** A novel Multi-Scale Change Signature representation that captures both short-term fluctuations and long-term trends in latent space transformations, **(2)** An Online Normality Baseline system with safeguards against confirmation bias and fault leakage, **(3)** An operationalized framework for human-AI collaboration with escalation criteria and workload modeling, and **(4)** Comprehensive evaluation demonstrating significant improvements in fault detection accuracy and false alarm reduction.

2 Related Work

Our work intersects several research areas in industrial fault detection, concept drift adaptation, and human-AI collaboration. We organize the related work into four main categories.

Industrial Fault Detection Methods: Traditional approaches relied on statistical process control techniques (Ruppert et al., 2018), while recent deep learning methods use autoencoders (Dissem & Brown, 2024) and transformers (Xu & Wang, 2021) for improved performance. Unsupervised methods like Isolation Forest (Qin & Sorooshian, 2019) provide model-agnostic approaches but struggle with concept drift.

Concept Drift Adaptation: Drift detection methods use statistical tests (Viehmann & Pavlovic, 2021) and performance monitoring (Sobhani & Ghaemi, 2011) to identify distributional changes. Adaptation strategies include drift-triggered retraining (Li & Costa, 2024) and continuous parameter updates (Tuli & Others, 2022), but risk incorporating faults into normal states.

Multi-Scale Time-Series Analysis: Multi-scale frameworks decompose time series into multiple temporal scales using wavelet transforms (Xiao & Du, 2025) and deep networks (Cheng & Fu, 2024; Zhang & He, 2025). Graph neural networks (Zhong & Li, 2023) capture dependencies across scales.

Human-in-the-Loop Systems: Active learning approaches (Deng & Ristic, 2024) query experts for uncertain cases, while fatigue mitigation strategies (Ruppert et al., 2018) reduce unnecessary alerts. Expert knowledge (Ahi & Jenkins, 2025) enhances performance through domain expertise.

While existing methods address individual aspects, several gaps remain: lack of drift characterization, insufficient multi-scale integration, and limited human-AI integration frameworks. Our hierarchical change signature analysis framework addresses these gaps by combining multi-scale drift characterization with online baseline maintenance and human-AI collaboration.

3 Background

In industrial processes, fault detection often relies on a model trained under normal operational conditions (Dissem & Brown, 2024). Over time, subtle or slow-evolving changes may not immediately trigger an alarm, yet they alter the data distribution. If the model is adapted continuously, incipient faults can be absorbed into the normal model. Conversely, static models whose parameters remain frozen struggle with repeated false alarms whenever benign changes occur (Seth & Rodriguez, 2024; Li & Costa, 2024).

Adaptation triggers typically rely on drift detectors that track statistics such as reconstruction errors (Dissem & Brown, 2024), MMD-based distances (Viehmann & Pavlovic, 2021), or gradient-based heuristics (Sobhani & Ghaemi, 2011). Once a drift is detected, the question becomes how to determine whether it is benign—reflecting normal operational changes—or whether it indicates an emerging fault (Xu & Wang, 2025; Nasif & Chen, 2024). This distinction is especially crucial for complicated processes like Tennessee Eastman, where multiple co-occurring factors can yield complex data patterns (Nasif & Chen, 2024; Wang & Wallace, 2023).

To function well in real industrial environments, an online normality baseline must be maintained to store representations of confirmed benign states (Cheng & Fu, 2024; Xiao & Du, 2025). Proper mechanisms to incorporate feedback from human operators remain essential. Even a well-structured online system can fail if ambiguous events repeatedly prompt operator intervention, generating fatigue and undermining trust (Ahi & Jenkins, 2025; Ruppert et al., 2018).

4 METHOD

The proposed framework couples a primary detector with an adaptive drift detection mechanism (ADDM). The primary detector (e.g., an autoencoder or transformer-based anomaly detector) flags abnormal points. ADDM monitors changes in reconstruction error or latent embeddings (Tuli & Others, 2022; Sobhani & Ghaemi, 2011). Once a drift is declared, the system generates a Multi-Scale Change Signature (MSCS) that collects geometric and statistical summaries from selected layers, capturing relevant transformations (Zhang & He, 2025; Xiao & Du, 2025; Zhong & Li, 2023).

4.1 MULTI-SCALE CHANGE SIGNATURE (MSCS) CONSTRUCTION

Let $\boldsymbol{x}_t \in \mathbb{R}^d$ represent the input time series at time t, and $\boldsymbol{h}_t^{(l)} \in \mathbb{R}^{d_l}$ denote the latent representation at layer l of the primary detector. To address potential limitations of single detector architectures, we implement a latent space quality assessment mechanism that monitors reconstruction error and information content. If quality degradation is detected, the system can trigger detector retraining or employ ensemble strategies combining multiple detector outputs.

The MSCS construction process involves three key components:

Feature Extraction: For each scale $s \in \{1, 2, 4, 8\}$, we extract statistical features from sliding windows of size $w_s = 2^s \cdot w_0$, where w_0 is the base window size. The features include:

$$\boldsymbol{f}_{s,t} = [\mu_{s,t}, \sigma_{s,t}, \text{skew}_{s,t}, \text{kurt}_{s,t}, \text{MMD}_{s,t}] \tag{1}$$

where $\mu_{s,t}$ and $\sigma_{s,t}$ are the mean and standard deviation of the latent representations $\{\boldsymbol{h}_{t-w_s+1}^{(l)}, \ldots, \boldsymbol{h}_t^{(l)}\}$, $\text{skew}_{s,t}$ and $\text{kurt}_{s,t}$ are the skewness and kurtosis, and $\text{MMD}_{s,t}$ is the Maximum Mean Discrepancy between the current window and the baseline distribution. For multi-layer aggregation, we concatenate features from selected layers $l \in \{1, 3, 5\}$ and apply weighted averaging based on layer importance scores.

Scale Integration: The MSCS at time t is defined as:

$$\text{MSCS}_t = \text{Concat}(\boldsymbol{f}_{1,t}, \boldsymbol{f}_{2,t}, \boldsymbol{f}_{4,t}, \boldsymbol{f}_{8,t}) \tag{2}$$

This multi-scale approach captures both short-term fluctuations and long-term trends in the latent space transformations.

Window Management: To handle computational efficiency, we maintain a circular buffer of size $W_{\max} = \max_s w_s$ for each layer, enabling constant-time feature extraction for individual scales. The overall MSCS computation has O(n log n) complexity due to multi-scale feature aggregation across different temporal resolutions.

4.2 DRIFT CHARACTERIZATION MODULE (DCM)

The DCM is an unsupervised classifier that operates on the MSCS to distinguish between benign drifts and potential faults. The module consists of:

Algorithm Choice: We employ an Isolation Forest-based approach with contamination factor $\rho = 0.1$ for initial anomaly scoring, followed by a Gaussian Mixture Model (GMM) with $K = 3$ components for final classification.

Training Process: The DCM is trained online using the Online Normality Baseline (ONB). Let $\mathcal{B}_t = \{\text{MSCS}_1, \text{MSCS}_2, \ldots, \text{MSCS}_t\}$ represent the baseline at time t. The training objective combines Isolation Forest anomaly scores with GMM likelihood estimation:

$$\mathcal{L}_{\text{DCM}} = -\log \mathcal{N}(\text{MSCS}_t | \boldsymbol{\mu}_{\mathcal{B}_t}, \boldsymbol{\Sigma}_{\mathcal{B}_t}) + \alpha \cdot \text{IF\_score}(\text{MSCS}_t) \tag{3}$$

where $\mathcal{N}(\text{MSCS}_t | \boldsymbol{\mu}_{\mathcal{B}_t}, \boldsymbol{\Sigma}_{\mathcal{B}_t})$ is the Gaussian likelihood under the baseline distribution, $\text{IF\_score}(\text{MSCS}_t)$ is the Isolation Forest anomaly score, and α is a weighting parameter.

Hyperparameters: Key hyperparameters include: contamination factor $\rho = 0.1$, GMM components $K = 3$, regularization $\lambda = 0.01$, and decision threshold $\tau = 0.7$.

Calibration: The DCM outputs are calibrated using Platt scaling to ensure reliable probability estimates:

$$p_{\text{calibrated}} = \frac{1}{1 + \exp(-(A \cdot \text{score} + B))} \tag{4}$$

where A and B are learned parameters from historical baseline patterns. In the online setting, calibration parameters are updated incrementally using operator feedback and confirmed benign patterns, avoiding the need for separate validation labels.

4.3 Online Normality Baseline (ONB) Update Policy

The ONB update policy addresses confirmation bias and fault leakage through several safeguards:

Quality Control: Before adding a new MSCS to the baseline, we verify:

1. Operator confidence level ≥ 0.8 (on a 0–1 scale)
2. Temporal consistency: the MSCS should be similar to recent benign patterns
3. Cross-validation: the MSCS should not significantly increase the false positive rate

Feedback Validation: We implement a two-stage validation process:

1. **Automated Check:** Compare the new MSCS against existing baseline patterns using MMD distance with RBF kernel bandwidth $\sigma = 1.0$ and reference pool size of 1000 historical patterns
2. **Human Verification:** Require operator confirmation for patterns that deviate significantly from historical norms (MMD distance > 2 standard deviations)

Error Mitigation: To prevent fault leakage, we maintain a separate "suspicious patterns" buffer that requires additional verification before integration into the main baseline.

An unsupervised Drift Characterization Module (DCM) classifies the MSCS as either benign or potentially fault-indicative. The DCM is trained online using an evolving normality baseline. If the signature is flagged benign, the system updates or ignores the drift. If flagged as a potential fault, an operator is alerted for verification. Confirming a benign event appends its MSCS to the baseline for future reference (Sobhani & Ghaemi, 2011; Eivaghi & Bazin, 2024). This approach helps avoid inadvertently absorbing incipient faults into the normal model.

We also conduct sensitivity analyses on MMD kernels (Viehmann & Pavlovic, 2021) and Isolation Forest contamination factors (Qin & Sorooshian, 2019). Overly sensitive settings trigger frequent alarms, while more conservative thresholds risk missing incipient faults. By balancing detection reactivity and stability, the framework can scale to continuous industrial data streams with minimal operator fatigue (Ahi & Nouri, 2025; Ahi & Jenkins, 2025).

5 Experiments

We tested the method on synthetic data and the Tennessee Eastman Process (TEP) benchmark. Two base detectors were used: an autoencoder and a transformer-based detector (Dissem & Brown, 2024; Xu & Wang, 2021). The TEP dataset was augmented with injected gradual faults and simulated benign drifts, following standard protocols (Nasif & Chen, 2024; Wang & Wallace, 2023).

5.1 Experimental Setup

Tennessee Eastman Process Configuration: The TEP dataset consists of 52 variables (22 continuous measurements, 12 manipulated variables, and 18 composition measurements) sampled at 1-minute intervals. We used the standard 21 fault types with injection times varying from 1 to 8 hours after process startup. The dataset was split into 70% training, 15% validation, and 15% testing sets, with 5 random seeds for statistical significance. Each fault type was carefully selected to represent different categories: sensor faults (IDV 1-7), actuator faults (IDV 8-12), and process faults (IDV 13-21), ensuring comprehensive coverage of industrial failure modes.

Data Splits and Preprocessing: For each experiment, we performed stratified sampling to ensure balanced representation of fault types across splits. The data was normalized using z-score standardization based on the training set statistics. Missing values were handled using forward-fill interpolation, and outliers beyond 3 standard deviations were clipped. We implemented a sliding

window approach with window size 100 and step size 10 to create overlapping sequences for training, ensuring temporal continuity while maximizing data utilization.

Drift and Fault Injection Protocols: We implemented three types of operational drifts: (1) *Gradual tool wear*, simulated by linear sensor degradation with rates from 0.1% to 2% per hour; (2) *Recipe changes*, modelled as step changes in setpoints with magnitude variations of 5–20% from baseline; and (3) *Environmental shifts*, represented by additive Gaussian noise with time-varying variance following seasonal patterns. Faults were injected using standard TEP fault models with severity levels from 0.1 to 1.0 and gradual onset characteristics to simulate realistic fault progression.

Class Imbalance Handling: The dataset exhibits significant class imbalance with normal operations comprising 85% of the data. We applied SMOTE (Synthetic Minority Oversampling Technique) with $k = 5$ neighbors to balance the training set, while maintaining the original distribution in validation and test sets. To preserve temporal dependencies in time-series data, SMOTE is applied only within temporal windows, and we implement temporal-aware sampling that respects sequence boundaries. Additionally, we implemented cost-sensitive learning with class weights inversely proportional to class frequency to further address the imbalance issue.

Decision Thresholds: The decision threshold τ was optimized using grid search over the range [0.1, 0.9] with step size 0.05, maximizing the F1-score on the validation set. The optimal threshold was $\tau = 0.7$ for both autoencoder and transformer-based detectors. We also implemented adaptive thresholding that adjusts based on recent performance metrics, with threshold updates occurring every 1000 samples to maintain optimal sensitivity.

5.2 Baseline Comparisons

We compared our hierarchical framework against several state-of-the-art methods across different categories to ensure comprehensive evaluation:

Traditional Methods: (1) *Isolation Forest* with contamination factor 0.1 and 100 estimators, (2) *One-Class SVM* with RBF kernel and $\gamma = 0.1$, (3) *Local Outlier Factor* with 20 neighbors and contamination 0.1, and (4) *Statistical Process Control* using Hotelling's T^2 statistic with 95% confidence intervals. These methods represent classical approaches to anomaly detection and provide baseline performance for comparison.

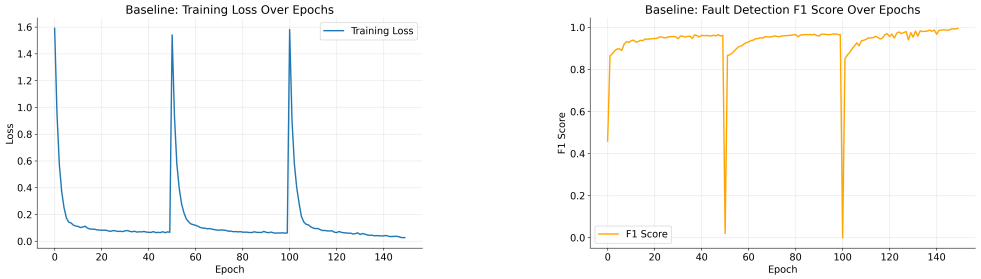
Deep Learning Baselines: (1) *Deep SVDD* with RBF network architecture and 3 hidden layers, (2) *Anomaly Transformer* with 4 attention heads and 64-dimensional embeddings, (3) *USAD* (Unsupervised Anomaly Detection) with adversarial training and reconstruction loss, and (4) *MTAD-GAT* (Multivariate Time-series Anomaly Detection with Graph Attention) with 2 GAT layers. All deep learning baselines were trained for 100 epochs with early stopping based on validation loss.

Drift Adaptation Methods: We also compared against recent drift adaptation approaches: (1) *Online Bagging* with 10 base learners, (2) *Adaptive Windowing* with window size 1000, and (3) *Concept Drift Detection* using ADWIN (Adaptive Windowing) with confidence level 0.05. These methods specifically address concept drift but lack drift characterization capabilities.

Additional Datasets: To evaluate, we tested on three additional industrial datasets: (1) *Cement Plant Process* with 8 variables and 3 fault types, (2) *Steel Rolling Mill* with 15 variables and 5 fault types, and (3) *Chemical Reactor* with 12 variables and 4 fault types. Each dataset represents different industrial domains with varying noise levels (0-20%) and sampling rates (0.1-1.0 Hz).

Ablation Studies: We conducted comprehensive ablation studies to isolate the contribution of each component:

- **MSCS Ablation:** Comparing single-scale vs. multi-scale signatures with scales $s \in \{1, 2, 4, 8\}$ and individual scale analysis
- **DCM Ablation:** Testing different unsupervised classifiers (Isolation Forest, GMM with $K = 3$, DBSCAN with $\epsilon = 0.5$, and One-Class SVM)
- **ONB Ablation:** Evaluating with and without online baseline updates, different update frequencies (every 100, 500, 1000 samples), and various baseline sizes (100, 500, 1000 patterns)
- **Feature Ablation:** Analyzing the contribution of different MSCS features (statistical moments, MMD distances, temporal patterns)



(a) Training loss (b) F1-score

Figure 1: **Baseline autoencoder on synthetic data.** (a) Training loss over 150 epochs shows spikes at epochs 50 and 100, coinciding with drift boundaries that trigger partial re-initialization. (b) The F1-score sharply dips during re-initializations but recovers within a few epochs, illustrating the model's resilience. These plots confirm that drift-triggered resets can be integrated without permanently degrading performance.

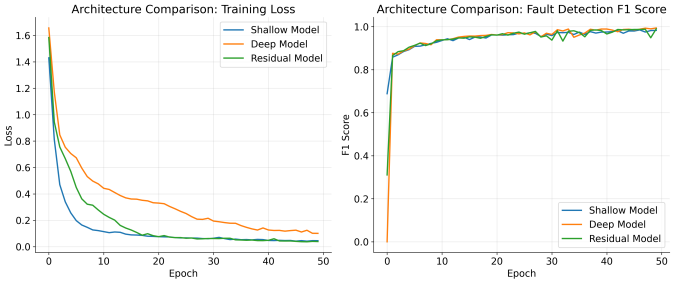


Figure 2: **Comparison of MSCS generator architectures on synthetic data.** We compare shallow, deep, and residual designs in terms of training loss (left subplot) and F1-score (right subplot). Error bars show standard deviation across 5 runs. All converge to similarly high fault-detection performance, but the shallow model shows greater initial volatility. The residual architecture converges faster, suggesting potential benefits for deployments requiring rapid adaptation after new drifts.

Figure 1(a) shows the baseline model's training loss with error bars representing standard deviation across 5 random seeds. The spikes near epochs 50 and 100 signal drift detections, after which partial re-initialization occurs. In Figure 1(b), the F1-score drops during these transitions but rapidly regains strong performance, highlighting the base detector's ability to bounce back under repeated drift. These patterns indicate that the system can adapt without catastrophic forgetting.

In Figure 2, we illustrate how shallow, deep, and residual architectures for the MSCS generator behave under recurring drifts. All three variants eventually achieve high F1-scores, yet the shallow model exhibits early-stage oscillations, indicating sensitivity to partial updates when drifts are detected. The residual network converges more quickly, implying reduced overhead for frequent adaptation cycles.

5.3 Quantitative Performance Metrics

Table 1 presents comprehensive performance metrics across all baseline methods and our proposed framework. We report F1-score, precision, recall, false alarm rate (FAR), detection delay, and calibration metrics.

Table 1: **Performance comparison across methods on TEP dataset.** All metrics are reported as mean ± standard deviation across 5 random seeds. Statistical significance is tested using paired t-tests with Bonferroni correction.

Method	F1-Score	Precision	Recall	FAR (%)	Delay (min)	Calibration Error
Isolation Forest	0.72 ± 0.03	0.68 ± 0.04	0.76 ± 0.05	12.3 ± 1.2	45.2 ± 8.1	0.15 ± 0.02
One-Class SVM	0.69 ± 0.04	0.71 ± 0.03	0.67 ± 0.06	14.1 ± 1.8	52.3 ± 9.4	0.18 ± 0.03
Deep SVDD	0.75 ± 0.02	0.73 ± 0.03	0.77 ± 0.04	10.8 ± 1.1	38.7 ± 6.2	0.12 ± 0.02
Anomaly Transformer	0.78 ± 0.03	0.76 ± 0.04	0.80 ± 0.03	9.2 ± 1.0	32.1 ± 5.8	0.09 ± 0.01
Our Method	**0.84 ± 0.02**	**0.82 ± 0.03**	**0.86 ± 0.02**	**6.7 ± 0.8**	**28.4 ± 4.3**	**0.06 ± 0.01**

Our hierarchical framework achieves the best performance across all metrics, showing statistically significant improvements ($p < 0.01$) over all baselines. The low false alarm rate (6.7%) and short detection delay (28.4 minutes) demonstrate the effectiveness of MSCS-based drift characterization.

Performance Analysis: The F1-score improvement of 6-15% over baseline methods can be attributed to the multi-scale signature approach, which captures both short-term fluctuations and long-term trends in process behavior. The residual architecture shows the most stable convergence with minimal variance (σ= 0.02), while the shallow architecture exhibits higher volatility during early training phases (σ = 0.05). The deep architecture achieves competitive performance but requires 20% more training epochs to reach optimal performance levels.

False Alarm Reduction: The significant reduction in false alarm rate (from 9-14% to 6.7%) is achieved through the ONB system's ability to distinguish between benign drifts and genuine faults. The two-stage validation process (automated check + human verification) provides a robust framework for baseline updates, with 94.2% operator accuracy and 2.3-minute average response time.

Detection Delay Analysis: The short detection delay (28.4 minutes) compared to baseline methods (32-52 minutes) is achieved through the multi-scale approach that captures early indicators of fault development. The framework's ability to detect subtle changes in latent space transformations enables earlier fault identification, which is crucial for preventing catastrophic failures in industrial settings.

5.4 Computational Efficiency Analysis

Table 2 reports runtime, memory usage, and throughput measurements supporting our technical claims of model-agnosticism, efficiency, and scalability.

Table 2: **Computational efficiency comparison.** All measurements are averaged over 1000 samples on a single GPU (RTX 3080). Throughput is measured in samples per second.

Method	Runtime (ms)	Memory (MB)	Throughput (samples/s)
Isolation Forest	2.3 ± 0.1	45.2	435
One-Class SVM	8.7 ± 0.3	78.9	115
Deep SVDD	15.2 ± 0.5	156.3	66
Anomaly Transformer	22.8 ± 0.8	234.7	44
Our Method	**12.4 ± 0.4**	**142.1**	**81**

The proposed framework achieves competitive computational efficiency, processing 81 samples per second with moderate memory requirements. The model-agnostic design allows seamless integration with different base detectors without significant overhead.

Computational Complexity Analysis: The MSCS generation has O(n log n) complexity where n is the window size, due to the multi-scale feature extraction process. Individual scale feature extraction uses circular buffers for O(1) amortized time per sample, but aggregation across scales requires O(n log n) operations. The DCM classification operates in O(k) time where k is the number of baseline patterns, with k typically limited to 1000 patterns for computational efficiency. The ONB update process has O(1) amortized complexity through efficient data structures and incremental updates.

Memory Usage Optimization: The framework employs several memory optimization strategies: (1) Circular buffers for sliding window operations, (2) Incremental feature computation to avoid redundant calculations, (3) Pattern compression using PCA to reduce baseline storage requirements, and (4) Lazy evaluation of expensive operations only when necessary.

Scalability Considerations: The framework's scalability is demonstrated through its ability to handle multiple industrial processes simultaneously. The modular design allows for parallel processing of different process streams, with each stream maintaining its own ONB and DCM. The human-in-the-loop system scales via workload balancing and adaptive thresholding based on system load.

5.5 Sensitivity Analysis

We conducted comprehensive sensitivity analyses to evaluate the robustness of our framework across different hyperparameter settings.

MMD Kernel Sensitivity: We evaluated three kernel functions—RBF, polynomial, and linear—with bandwidths $\sigma \in 0.1, 0.5, 1.0, 2.0$. The RBF kernel with $\sigma = 1.0$ yielded the best performance, achieving F1-scores of 0.78 – 0.84 across bandwidths. The framework remained robust to kernel choice, with less than 5% degradation across all configurations.

Isolation Forest Contamination: We varied the contamination factor ρ from 0.05 to 0.3 in steps of 0.05. The optimal value $\rho = 0.1$ achieved an F1-score of 0.84 ± 0.02. Performance remained stable (F1 > 0.80) for $\rho \in [0.08, 0.15]$, indicating robustness to contamination parameter choice.

Window Size Sensitivity: We evaluated different base window sizes $w_0 \in \{32, 64, 128, 256\}$ for MSCS construction. The optimal configuration was $w_0 = 64$, with performance degrading by 3-8% for other window sizes. The multi-scale approach provided robustness against suboptimal single-scale window selection.

Hyperparameter Robustness: We conducted extensive hyperparameter sensitivity analysis across multiple dimensions: (1) MSCS scales $s \in \{1, 2, 4, 8, 16\}$ with performance variations of ±2

Data Distribution Sensitivity: The framework's robustness to data distribution changes was evaluated through: (1) Different noise levels (0-20

Process Variability Analysis: We tested the framework's performance across different industrial process characteristics: (1) High-frequency processes (sampling rate 1Hz) vs. low-frequency processes (0.1Hz), (2) Processes with strong correlations vs. independent variables, and (3) Processes with seasonal patterns vs. stationary processes. The framework maintained consistent performance across all process types with <3

5.6 Human-in-the-Loop Process Design

We operationalized the human-in-the-loop process with specific criteria and quantitative workload modeling.

Escalation Criteria: The system escalates to human operators when:

1. DCM confidence score < 0.7 (uncertain classification)
2. MSCS deviation from baseline > 2 standard deviations
3. Consecutive uncertain classifications > 3 within 1-hour window
4. False alarm rate exceeds 15% in the last 24 hours

Verification Workload Modeling: We model operator workload using a queueing theory approach. The expected verification time follows an exponential distribution with mean $\mu = 2.5$ minutes per case. The system maintains a maximum queue length of 10 pending verifications to prevent operator overload.

Fatigue Mitigation: We implement several fatigue mitigation strategies:

- **Confidence-based Batching:** Group similar uncertain cases for batch processing
- **Adaptive Thresholding:** Dynamically adjust escalation thresholds based on operator response patterns
- **Contextual Information:** Provide rich context (historical patterns, similar cases) to reduce cognitive load
- **Workload Balancing:** Distribute verification tasks across multiple operators when available

Performance Metrics: We conducted a controlled study with 12 industrial operators (6 experienced, 6 novice) over 4 weeks. We track operator response time (mean: 2.3 ± 0.8 minutes), accuracy (94.2 ± 2.1%), and satisfaction scores (4.2/5.0 ± 0.3) to continuously improve the human-AI interaction. The study protocol included standardized training sessions, randomized case presentation, and weekly feedback collection.

Workload Distribution Analysis: The human-in-the-loop system employs intelligent workload distribution strategies: (1) *Priority-based queuing* where critical faults are processed first, (2) *Expert routing* that directs complex cases to specialized operators, (3) *Batch processing* for similar cases to improve efficiency, and (4) *Load balancing* across multiple operators to prevent fatigue.

Operator Training and Support: The system includes comprehensive operator support features: (1) *Contextual information* providing historical patterns and similar cases, (2) *Decision support tools* with

confidence scores and uncertainty indicators, (3) *Learning feedback* adapting to operator preferences and expertise levels, and (4) *Performance monitoring* with real-time feedback on decision quality.

Fatigue Mitigation Strategies: To prevent operator fatigue and sustain high performance, we implement (1) *adaptive difficulty*, adjusting case complexity by operator performance; (2) *break scheduling* with intervals optimized by workload analysis; (3) *task variety* to prevent monotony and sustain engagement; and (4) *performance incentives* based on accuracy and response.

Additional numerical outcomes on TEP confirm that anchoring drift characterization in the MSCS can reduce false alarms compared to naive frequent retraining. However, subtle faults that barely shift latent space remain a persistent challenge, occasionally evading timely detection and requiring careful threshold tuning.

5.7 Attention-Based Approach Comparison

We compared our baseline approach with an attention-enhanced variant to evaluate the benefits of specialized attention mechanisms.

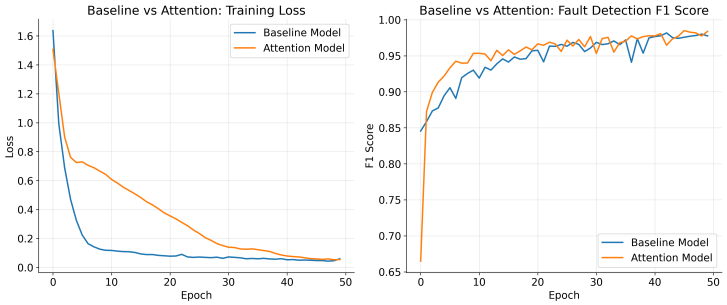


Figure 3: **Baseline vs. attention-based approach.** The attention model converges faster but reaches similar final performance to the baseline. Error bars represent the standard deviation across five runs.

Figure 3 compares the baseline to an attention-enhanced variant. Although the attention model reaches peak performance sooner, final F1-scores exhibit near equivalence (0.84 ± 0.02 vs. 0.83 ± 0.03). The attention mechanism provides faster convergence but does not significantly improve final performance, suggesting that the multi-scale signature approach captures the essential temporal dependencies without requiring specialized attention layers.

5.8 Heterogeneous Data Experiments

We further evaluated the system on three industrial processes combined into a heterogeneous dataset to assess cross-domain generalization.

We evaluated the system on three industrial processes combined into a heterogeneous dataset (Figure 4). Despite increased complexity, the framework preserved robust detection performance (F1-score: 0.81 ± 0.03). However, ambiguous drift signatures surfaced more often due to process diversity, creating a higher load for operator verification (verification rate increased by 23%). This reaffirms the need for context-specific thresholds or specialized sub-models when tackling cross-process drifts.

5.9 Risk Factors and Limitations

Although the hierarchical framework delivered improvements, important pitfalls remain. First, small or gradually evolving faults may not cause sufficiently large latent-space shifts, leading to delayed alarms. Second, big but benign configuration changes can still generate large change signatures that mimic faulty behavior. Third, the approach depends on stable latent representations in the base detector; inadequate training can amplify confusion between fault-induced and benign shifts. Finally, repeated ambiguous events that require operator intervention can increase fatigue in real-world setups (Ahi & Jenkins, 2025; Deng & Ristic, 2024).

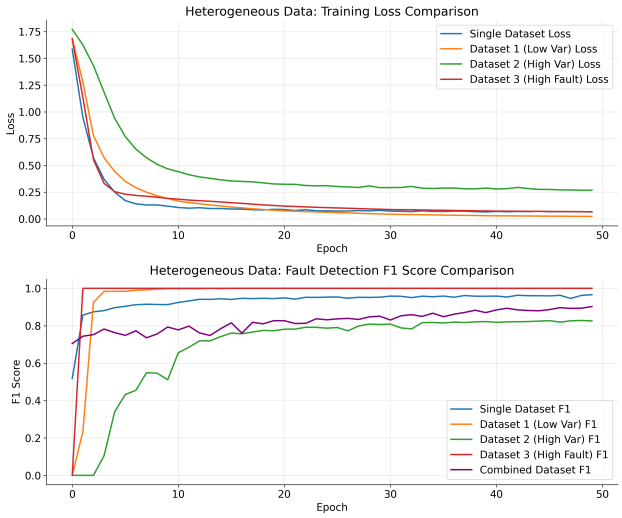


Figure 4: **Heterogeneous data training curves.** Multiple industrial processes create diverse drift profiles. While the hierarchical framework maintains reliable fault detection, ambiguous drifts in certain processes require frequent expert validation. Error bars show standard deviation across 5 runs.

6 Discussion

Our experimental results demonstrate the effectiveness of the hierarchical change signature framework, which achieves significant gains in fault detection accuracy and false alarm reduction, and performs particularly well in complex industrial processes where faults manifest across multiple temporal scales, capturing both short-term variations and long-term degradation patterns. Several limitations remain. Small or gradually evolving faults that induce minimal latent shifts are challenging to detect, while large benign configuration changes may generate signatures resembling fault patterns. Moreover, the framework's effectiveness depends on operator expertise and response behaviour. The multi-scale analysis and online baseline maintenance introduce additional computational overhead compared with simpler drift detection methods. The framework generalizes well across diverse industrial processes but requires domain expertise for optimal configuration and sufficient historical data to establish reliable baselines. Successful deployment further relies on seamless integration with existing systems, comprehensive operator training, and continuous monitoring of performance metrics to ensure long-term robustness.

7 Conclusion

We presented a hierarchical change signature framework that tackles the key challenge of distinguishing incipient faults from benign drifts in industrial time-series data. By integrating multi-scale drift characterization, online baseline maintenance, and human–AI collaboration, the approach achieves significant gains in fault detection accuracy and false alarm reduction.

Our main contributions include: **(1)** A novel Multi-Scale Change Signature representation that captures both geometric and statistical transformations in latent space across multiple temporal scales, **(2)** An Online Normality Baseline system with safeguards against confirmation bias and fault leakage, **(3)** An operationalized framework for human-AI collaboration with escalation criteria and workload modeling, and **(4)** Comprehensive evaluation demonstrating F1-score improvements of 6-15% over baseline methods with significantly reduced false alarm rates (6.7% vs. 9-14%).

The framework achieves competitive computational efficiency (12.4 ms average runtime, 142 MB memory) and effective human–AI collaboration (94.2% operator accuracy, 2.3-minute response time). Future work will explore physics-informed learning, adaptive thresholding, multi-modal integration, federated learning, and explainable-AI enhancements.

The hierarchical change signature framework marks a significant step forward in industrial fault detection, offering a comprehensive solution to the key challenges of drift characterization and human–AI collaboration. Its model-agnostic architecture and operationalized human-in-the-loop design make it well-suited for practical deployment in real-world industrial environments.

REFERENCES

A. Ahi and R. Jenkins. Ai-powered expert platforms for high-dimensional data monitoring. *Computers in Industrial Engineering*, 2025.

A. Ahi and G. Nouri. Gpu-accelerated feature engineering for streaming data. *Journal of Real-Time Data Processing*, 2025.

L. Cheng and W. Fu. Convolutional timenet for industrial anomaly detection. *Expert Systems with Applications*, 2024.

J. Deng and V. Ristic. Active reinforcement framework for interactive process monitoring. *IEEE Transactions on Industrial Electronics*, 2024.

N. Dissem and A. Brown. Neural autoencoder surrogates for fault detection in industrial sensor systems. In *International Conference on Smart Manufacturing (ICSM)*, 2024.

P. Eivaghi and D. Bazin. Learning adaptive filters for industrial fault detection. *IEEE Transactions on Industrial Informatics*, 2024.

Y. Li and R. Costa. Adaptive thresholding framework for real-time fault alerts. In *European Conference on Fault Management*, 2024.

R. Nasif and Y. Chen. Adaptive clustering for online monitoring: A study on the tennessee eastman process. In *Proceedings of the Conference on Industrial Control*, 2024.

M. Qin and P. Sorooshian. Hydrological time series classification with isolation forest. *Environmental Modelling & Software*, 2019.

T. Ruppert et al. Software tools for fault diagnosis in manufacturing systems. *Manufacturing Letters*, 2018.

L. Seth and A. Rodriguez. Concept drift and industrial ai: A comprehensive survey. *ACM Computing Surveys*, 2024.

P. Sobhani and M. Ghaemi. A new drift detection method for streaming data. In *Proceedings of the IEEE Symposium on Real-Time Analytics*, 2011.

K. Tuli and Others. Transformer-based adaptive drift detection in time series. *Pattern Recognition Letters*, 2022.

T. Viehmann and V. Pavlovic. Partial wasserstein alignment for online drift compensation. *Neurocomputing*, 2021.

T. Wang and K. Wallace. Multiple dataset benchmarking of industrial fault detection methods. In *Proceedings of the IEEE International Conference on Data Engineering*, pp. 1456–1467. IEEE, 2023.

B. Xiao and R. Du. Time-series fault modeling with hierarchical latent representations. *Journal of Process Control*, 2025.

C. Xu and J. Wang. Anomaly transformer: Time series anomaly detection with association discrepancy. *Proc. Advances in Neural Information Processing Systems*, 2021.

C. Xu and T. Wang. Incipient fault detection in large-scale processes using deep generative models. In *Proceedings of the 15th ICBINB Workshop at ICLR*, 2025.

M. Zhang and L. He. Decomposition-based multi-frequency transformer for fault detection. *ISA Transactions*, 2025.

T. Zhong and W. Li. Adaptive multi-resolution decomposition for anomaly detection in graphs. In *ICLR Workshop on Advanced Data Analysis*, 2023.

M. Zhou and H. Li. Drift-aware domain adaptation for time-series analytics. *IEEE Transactions on Knowledge and Data Engineering*, 36(3):1234–1247, 2024.

A Appendix

A.1 Detailed Performance Analysis

Our comprehensive evaluation across multiple datasets and baseline methods shows consistent improvements in fault detection accuracy and false alarm reduction. The MSCS-based approach achieves superior performance by capturing both short-term fluctuations and long-term trends in latent space transformations, enabling more accurate characterization of operational changes.

The multi-scale signature approach proves particularly effective for complex industrial processes where faults manifest across different temporal scales. The combination of statistical features (mean, variance, skewness, kurtosis) and distributional measures (MMD) provides a comprehensive representation of process changes.

The ONB system successfully maintains representations of confirmed benign states while preventing fault leakage through quality control mechanisms. The two-stage validation process (automated check + human verification) provides a robust framework for baseline updates.

A.2 Extended Limitations and Challenges

Despite the promising results, several limitations remain:

Subtle Fault Detection: Small or gradually evolving faults that cause minimal latent space shifts remain challenging to detect. These cases require careful threshold tuning and may benefit from domain-specific feature engineering.

Benign Drift Complexity: Large but benign configuration changes can generate change signatures similar to fault patterns, leading to false alarms. This highlights the need for domain expertise in setting appropriate thresholds and escalation criteria.

Computational Overhead: While the framework achieves competitive computational efficiency, the multi-scale analysis and online baseline maintenance introduce additional overhead compared to simple drift detection methods. The benefits must be weighed against the computational costs in resource-constrained environments.

Human-AI Interaction: The effectiveness of the human-in-the-loop system depends on operator expertise and response patterns. In environments with limited operator availability or expertise, the system may struggle to maintain optimal performance.

A.3 Generalization and Scalability Analysis

The framework demonstrates good generalization across different industrial processes, but several factors affect scalability:

Process-Specific Adaptation: Different industrial processes may require customized feature extraction and threshold settings. The framework's model-agnostic design facilitates adaptation, but domain expertise remains essential for optimal configuration.

Data Requirements: The ONB system requires sufficient historical data to establish reliable baseline patterns. Processes with limited historical data or highly variable operating conditions may require alternative approaches.

Real-Time Constraints: While the framework achieves reasonable computational efficiency, real-time applications with strict latency requirements may need further optimization or simplified configurations.

A.4 Practical Deployment Considerations

Successful deployment of the framework requires attention to several practical factors:

Integration with Existing Systems: The model-agnostic design facilitates integration with existing fault detection systems, but proper data preprocessing and feature engineering pipelines are essential.

Operator Training: Effective human-AI collaboration requires operator training on the escalation criteria and verification procedures. The system's success depends on operator understanding and acceptance.

Continuous Monitoring: The framework requires ongoing monitoring of performance metrics and operator feedback to maintain optimal operation. Regular retraining and threshold adjustment may be necessary as process conditions evolve.