Applied Soft Computing Journal 184 (2025) 113761

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Real-time anomaly detection in seasonal time series with conditional variational autoencoder

Lorenzo Porcelli [a],*, Marcello Trovati [b], Francesco Palmieri [a]

[a] Department of Computer Science, University of Salerno, Fisciano, SA, Italy
[b] Department of Computer Science, Edge Hill University, Ormskirk L39 4QP, UK

ARTICLE INFO

Dataset link: https://bit.ly/3Jsh3dv

Keywords:
Anomaly detection
Variational autoencoders
Time series
Seasonal data

ABSTRACT

Real-time anomaly detection in high-frequency seasonal time series is commonly addressed using prediction-based methods, which require waiting for new values to perform subsequent predictions and demand continuous processing over time. This work introduces a novel framework for real-time anomaly detection in seasonal time series, with a practical implementation using Conditional Variational Autoencoders based on Multilayer Perceptrons. Our approach eliminates the need for historical time series data at inference time, instead generating a one-shot long-term expected time series that enables immediate evaluation of streaming data with minimal computational resources. Empirical evaluations on real-world seasonal time series demonstrate that the proposed approach achieves state-of-the-art performance compared in both semi-supervised and unsupervised settings. The framework provides computational efficiency and low energy consumption, making it suitable for deployment in commodity hardware and offline environments.

1. Introduction

Recent advancements in computational and storage capabilities, together with the explosion of big data availability, have significantly highlighted the significance of time series analysis, particularly in anomaly detection. Anomaly detection, which identifies data instances deviating from expected behavior [1], has become critical across various sectors, including finance, manufacturing, healthcare, cybersecurity, and earth sciences.

Time series data can be categorized as univariate (UTS) or multivariate (MTS). UTS involve a single variable that changes over time, such as temperature readings every 30 min. Conversely, MTS encompass multiple variables per data point, exemplified by simultaneous recordings of temperature, humidity, and wind. The proliferation of sensors has led to higher-dimensional data, often characterized by noise, non-stationarity, and seasonality. These factors complicate the distinction between true anomalies and natural variations, making the anomaly detection task more challenging.

Anomaly categorization varies across domains, but three primary patterns predominate: point, collective, and contextual anomalies. Point anomalies are individual data instances that significantly deviate from the baseline normal pattern, thus appearing abnormal when examined in isolation. Collective anomalies, in contrast, emerge from a group of related data points that, while potentially normal individually, exhibit anomalous behavior when considered collectively. Contextual anomalies are data instances that deviate from the norm only within specific contexts, necessitating consideration of both contextual and behavioral attributes for identification. The contextual nature of these anomalies implies that what constitutes normal behavior in one setting may be anomalous in another.

Time series anomaly detection methods can be broadly categorized into three principal approaches: prediction-based, proximity-based, and reconstruction-based. Prediction-based methods utilize predictive models to forecast future values in a time series, identifying anomalies when observed values significantly deviate from predictions. Common techniques include autoregressive models, recurrent neural networks (RNN, LSTM), and linear regression. Proximity-based methods detect anomalies by measuring the distance between data points, flagging those significantly distant from their neighbors as anomalous. Key techniques include K-Nearest Neighbors (KNN) and DBSCAN. Reconstruction-based methods employ learning models to reconstruct time series values, identifying anomalies when there are substantial differences between the original and reconstructed values. Autoencoders, Principal Component Analysis (PCA), and clustering models are frequently used in this approach.

Prediction-based and proximity-based methods are generally capable of rapid inference. Proximity-based approaches, such as KNN, do not require a temporal window for inference, but may be less effective

* Corresponding author.
E-mail addresses: lporcelli@unisa.it (L. Porcelli), Marcello.Trovati@edgehill.ac.uk (M. Trovati), fpalmieri@unisa.it (F. Palmieri).

https://doi.org/10.1016/j.asoc.2025.113761
Received 27 October 2024; Received in revised form 21 July 2025; Accepted 17 August 2025
Available online 22 August 2025

with complex or high-dimensional data. Reconstruction-based methods, while potentially more robust for complex data, often have slower inference times and are therefore less suitable for high-frequency time series.

We introduce a novel approach for real-time anomaly detection (RTAD) in seasonal time series, employing a conditional variational autoencoder (cVAE) with a Multilayer Perceptron architecture. Unlike conventional sequential prediction-based methods that depend on historical data during inference, the proposed RTAD-cVAE generates new time series data by sampling from a conditional probability distribution based on time-dependent conditions.

The proposed approach offers enhanced performance over existing methods in handling seasonal time series: (1) it enables real-time anomaly detection; (2) it is energy efficient in high-frequency data streams; and (3) it exhibits robustness to partial or missing data.

The rest of this article is structured as follows. Section 2 provides a comprehensive review of anomaly detection approaches, categorizing them into proximity-based, reconstruction-based, and prediction-based methods, and contextualizing our proposed method within the existing landscape of techniques. Section 3 outlines the proposed real-time anomaly detection framework based on conditional variational autoencoders, detailing the helical encoding, the timestamp trick, and the complete architecture for seasonal time series anomaly detection. Section 4 describes the experimental setup, including datasets, baseline methods, evaluation criteria, and implementation details for comprehensive performance assessment. Section 5 presents quantitative comparisons with state-of-the-art techniques across multiple datasets, including ablation studies, sensitivity analyses, robustness evaluations, scalability evaluations, and energy efficiency measurements. Section 6 discusses the performance characteristics, computational advantages, operational benefits, and limitations identified through our comprehensive experimental evaluation. Finally, Section 7 concludes this work and outlines the directions for future research.

2. Related work

Anomaly detection approaches for time series data can be classified according to their training paradigms as supervised, semi-supervised or unsupervised. In practical applications, supervised training faces significant limitations due to the inherent scarcity of labeled data and its ineffectiveness against previously unknown anomalous phenomena. Consequently, the proposed model is compared with the models presented in Table 1 under unsupervised or semi-supervised paradigms. Unsupervised training involves training on data that contains both normal and anomalous observations (unlabeled). Semi-supervised training, also known as novelty detection, utilizes only normal-class data during training to establish a baseline of typical behavior and evaluates new observations against this learned baseline.

2.1. Proximity and isolation approaches

Proximity and isolation approaches identify anomalies through their distance from normal data patterns or their ease of isolation, operating on the assumption that anomalies exhibit significant deviation from typical observations in the feature space.

The one-class support vector machine (OCSVM) [2] is an outlier detection method derived from Support Vector Machines, typically trained in a semi-supervised manner using only normal data to define a boundary around normal observations. Isolation Forest (iForest) [3] applies the random forest concept to isolate outliers and can be trained with unsupervised learning on mixed data, requiring a preset contamination ratio. Unlike many methods that first profile normal behavior, iForest directly identifies anomalies by exploiting the fact that anomalies are typically easier to isolate than normal instances.

Median Absolute Deviation (MAD-AD) [4] provides a statistical framework for anomaly detection by calculating deviations from the median rather than the mean. This approach offers enhanced robustness to outliers by measuring distances between data points and the median in terms of median distance itself, making it particularly effective for handling skewed distributions where mean-based methods might fail.

Deep Isolation Forest (DIF) [5] enhances the traditional isolation forest concept by incorporating deep learning techniques. By employing deep representation ensembles, it achieves non-linear isolation capabilities in the original data space that surpass conventional tree-based approaches. This method bridges the gap between traditional isolation techniques and modern neural representations, resulting in improved detection performance for complex anomaly patterns.

Calibrated One-class classifier for Unsupervised Time Series Anomaly detection (COUTA) [6] introduces a neural network architecture specifically designed for anomaly detection in time series data. COUTA learns comprehensive representations of normal patterns and incorporates a calibration mechanism to establish more reliable decision boundaries. This approach effectively identifies anomalies as deviations from learned normality patterns while minimizing detection threshold sensitivity issues common in other methods.

Skyline [7] represents an ensemble approach that combines multiple expert detectors to provide real-time anomaly detection by aggregating anomaly scores from diverse methods, leveraging the strengths of different proximity-based techniques for improved detection performance.

2.2. Reconstruction-based approaches

Reconstruction-based methods learn a low-dimensional latent representation of the data to reconstruct the original input. These approaches operate on the assumption that anomalies, being rare occurrences, are not effectively captured in the latent space mapping.

Principal Component Analysis (PCA) [8], while effective for dimensionality reduction, suffers from limitations in handling non-linear relationships and struggles with spatial–temporal correlations in multivariate settings. Autoencoder methods, such as LSTM-AE [9], use AE to learn effective latent space representations, although they require careful calibration of latent space dimensionality to balance between meaningful pattern generalization and unwanted noise incorporation.

Generative methods, including Variational Autoencoders (VAE) [10] and Generative Adversarial Networks (GAN), address overfitting challenges through different mechanisms. VAE approaches such as LSTM-VAE [11] introduce regularization into the latent space through probabilistic encoders and decoders. GANs apply regularization to reconstruction errors through adversarial learning frameworks. MAD-GAN [12] effectively captures non-linear latent interactions by leveraging spatial–temporal correlations. TadGAN [13] enhances stability by incorporating cycle consistency loss during training, mitigating the overfitting problems common in traditional reconstruction methods. BeatGAN [14] further refines these approaches by focusing specifically on detecting anomalous beats or segments within time series through specialized adversarial reconstruction networks.

The Auto-encoder with Regression (AER) [15] represents a hybrid approach that extends reconstruction-based methods by combining an auto-encoder focused on reconstruction with a regression component aimed at prediction. By employing a joint objective function, AER generates both reconstruction-based and prediction-based anomaly scores concurrently, harnessing the benefits of both methodological families.

Transformer-based architectures offer a computationally efficient alternative to LSTM-based models commonly used in VAE and GAN approaches for time series, leveraging self-attention mechanisms for temporal modeling. Anomaly Transformer [16] utilizes Anomaly-Attention to calculate Association Discrepancy between prior-association (bias towards adjacent anomalies) and series-association (global associations for normal points), combining reconstruction loss with normalized association discrepancy for anomaly scoring. TranAD [17] employs

Table 1
Overview of baseline anomaly detection methods compared with the proposed approach. Training mode indicates the composition of the training dataset: semi-supervised methods use normal data only, while unsupervised methods use mixed normal and anomalous data during training.

Ref.	Method	Dimensionality	Approach	Training mode
[19]	MA	Univariate	Statistical	Semi-supervised
[20]	SIMA	Univariate	Statistical	Semi-supervised
[19]	SARIMA	Univariate	Statistical	Semi-supervised
[20]	MA+SARIMA	Univariate	Statistical	Semi-supervised
[20]	SIMA+SARIMA	Univariate	Statistical	Semi-supervised
[21]	Multi-SARIMA	Univariate	Statistical	Semi-supervised
[21]	MA+Multi-SARIMA	Univariate	Statistical	Semi-supervised
[21]	SIMA+Multi-SARIMA	Univariate	Statistical	Semi-supervised
[22]	TBATS	Univariate	Statistical	Semi-supervised
[23]	Twitter ADVec	Univariate	Statistical	Semi-supervised
[7]	Skyline	Univariate	Machine learning	Semi-supervised
[24]	Numenta	Univariate	Machine learning	Semi-supervised
[25]	NumentaTM	Univariate	Machine learning	Semi-supervised
[16]	Anomaly Transformer	Multivariate	Deep learning	Semi-supervised
[17]	TranAD	Multivariate	Deep learning	Semi-supervised
[18]	TimesNet	Multivariate	Deep learning	Semi-supervised
[26]	DeepAnT	Multivariate	Deep learning	Semi-supervised
[27]	Piecewise AD	Univariate	Statistical	Unsupervised
[4]	MAD-AD	Univariate	Statistical	Unsupervised
[28]	Prophet	Univariate	Statistical	Unsupervised
[2]	OCSVM	Multivariate	Machine learning	Unsupervised
[3]	iForest	Multivariate	Machine learning	Unsupervised
[29]	LSTM-FD	Multivariate	Deep learning	Unsupervised
[30]	LSTM-AD	Multivariate	Deep learning	Unsupervised
[31]	AD-LTI	Multivariate	Deep learning	Unsupervised
[13]	TadGAN	Multivariate	Deep learning	Unsupervised
[6]	COUTA	Multivariate	Deep learning	Unsupervised
[5]	DIF	Multivariate	Deep learning	Unsupervised
[15]	AER	Multivariate	Deep learning	Unsupervised

attention-based sequence encoders to capture temporal trends while incorporating focus score-based self-conditioning and adversarial training to amplify reconstruction errors and improve stability.

However, vanilla attention mechanisms may struggle with anomaly detection as they calculate similarity between all temporal point pairs, potentially being dominated by prevalent normal patterns. To address this limitation, TimesNet [18] transforms temporal data into 2D representations based on periodicity, enabling the application of 2D convolutional kernels to model intricate temporal variations more effectively while maintaining the benefits of attention-based processing.

2.3. Prediction-based approaches

While generative models aim to solve the more general problem of learning a joint distribution over all variables, providing a deeper understanding of the data generation process, discriminative models focus on learning a predictor based on observations. Prediction-based methods for anomaly detection forecast future data points based on past patterns, identifying anomalies when substantial differences occur between predicted and actual values.

Moving Average (MA) represents a time series as deviations from the mean, though it can produce false positives by assuming normally distributed data values. Seasonal Integrated Moving Average (SIMA) models extend MA by incorporating seasonal differencing to account for cyclic trends. Seasonal Autoregressive Integrated Moving Average (SARIMA) models [32] offer more complexity to handle both trends and seasonality, though with increased computational requirements.

Multi-SARIMA [21] extends SARIMA by incorporating multiple seasonal components, allowing it to model datasets with two or more seasonal trends. TBATS (Trigonometric seasonality, Box–Cox transformation, ARMA errors, Trend and Seasonal components) [22] is a multi-seasonal forecasting model that handles complex seasonal patterns, including non-integer and high-frequency seasonality. It uses Fourier representations and ARMA error correction, making it effective for datasets with multiple seasonalities.

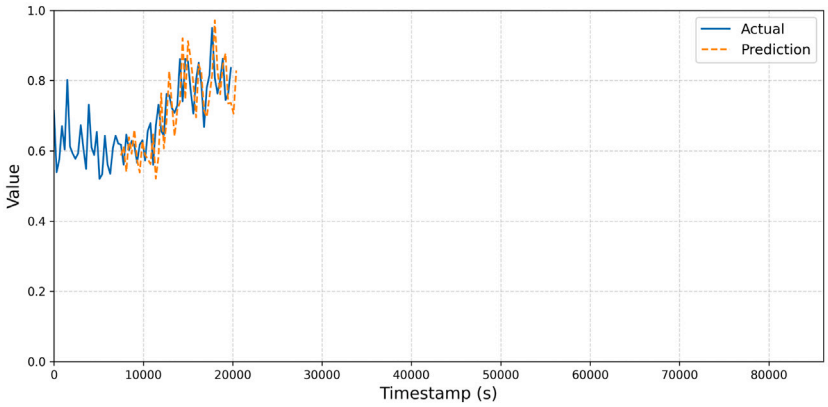
Prophet [28] employs an additive regression model that fits nonlinear trends with multiple seasonal components, showing effectiveness in business forecasting applications with strong seasonal patterns. It works particularly well with time series data that exhibit daily, weekly, and yearly seasonality patterns, along with holiday effects. Piecewise Median Anomaly Detection (Piecewise AD) [27] processes time series data by dividing it into fixed-size windows and detects anomalies using a decomposable series model, making it effective for handling local context variations. Twitter's Anomaly Detection (Twitter ADVec) [23] is specifically designed to target seasonal anomalies in social network data, automatically computing thresholds for anomaly detection in periodic time series with strong seasonal components.

Hybrid statistical approaches like MA+SARIMA [20] and MA+Multi-SARIMA [21] combine the strengths of different models in a two-step process. The faster MA model provides initial anomaly labels, while the more accurate but computationally intensive SARIMA model verifies only those data points flagged as potential anomalies. This approach balances speed and accuracy by using the simpler model to reduce the workload of the more complex one.

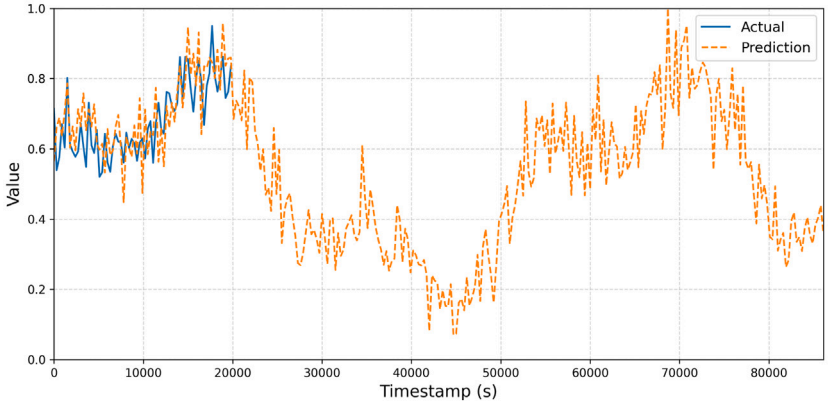
Deep Learning prediction-based methods generally require less domain knowledge than statistical models but may result in more false detections compared to reconstruction methods [15]. LSTM-based approaches include LSTM-FD [29], a prediction-driven approach leveraging a frame-to-frame Long Short-Term Memory network that identifies anomalies through analysis of prediction error distributions, and LSTM-AD [30], which employs a multi-source prediction scheme for anomaly detection. LSTM-DT [33] employs separate LSTM networks for each channel, reducing errors in high-dimensional outputs.

DeepAnT [26] implements a deep learning approach using Convolutional Neural Networks for time series anomaly detection, with separate modules for prediction and anomaly detection. AD-LTI [31] calculates Local Trend Inconsistency to assess the deviation of a data point from predictions generated from multiple previous frames.

Numenta [24] implements Hierarchical Temporal Memory (HTM), a biologically-inspired sequence memory algorithm for online anomaly detection. HTM learns patterns in streaming data and automatically



(a)



(b)

Fig. 1. Comparison of prediction-based anomaly detection approaches on high-frequency time series. (a) Sliding-window-based methods require continuous retraining on recent historical data to generate new forecasts. These methods typically predict only a few steps ahead to avoid error accumulation, necessitating frequent model updates as new observations become available. (b) The proposed approach generates long-term predictions (e.g. daily patterns) without requiring recent historical observations, enabling immediate anomaly detection from the start of monitoring.

adapts to evolving data distributions. The algorithm predicts future values and identifies anomalies by comparing these predictions against actual observations. Building on this foundation, NumentaTM [25] extends the approach to simultaneously detect both spatial and temporal anomalies within continuous data streams.

Most deep learning anomaly detection methods for time series use sliding windows to transform sequential data into supervised learning problems. These methods predict future values based on historical observations within fixed-size windows, requiring careful tuning of window size and prediction horizon. In real-time applications, sliding window methods introduce detection lag as they must wait for sufficient historical data before making predictions. In the presence of missing data, the predictive accuracy can degrade significantly until the window stabilizes again.

For seasonal time series, we leverage the timestamp trick [34] to eliminate historical data dependency at inference. Our approach generates long-term forecasts by sampling from a learned latent distribution conditioned on seasonal context, enabling real-time anomaly detection by comparing generated patterns with streaming data regardless of missing observations. Fig. 1 compares our generation-based method (Fig. 1(b)) with sliding window approaches (Fig. 1(a)). Although sliding window methods require warm-up periods and optimal window size selection, our approach provides immediate anomaly detection without these constraints. The following section provides a detailed explanation of the proposed approach.

3. Prediction-based anomaly detection with the timestamp trick

We address the problem of real-time anomaly detection in high-frequency seasonal time series under the following constraints: independence from historical data at inference time, robustness to partial or missing streaming data, computational efficiency for commodity hardware, and offline operation capability. Our solution leverages a framework based on helical encoding and the timestamp trick from [34], which transforms sequential forecasting into a conditional generation problem. This section presents our implementation using a conditional variational autoencoder for anomaly detection.

3.1. Data preprocessing

The helical encoding [34] provides an explicit geometric representation of seasonal patterns in time series data by positioning each observation on a helical trajectory. This encoding enables models to learn additional temporal coordinates that, in turn, enable ordering relationships to be established among independent outputs of a generative model. We implement a simplified version of the helical encoding, using only the xy-coordinates of the helix.

Consider a multivariate time series with n features and length T:

$$\{\mathbf{x}_t\}_{t=1}^{T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\} = \left\{ \begin{bmatrix} x_{1,1} \\ x_{1,2} \\ \vdots \\ x_{1,n} \end{bmatrix}, \begin{bmatrix} x_{2,1} \\ x_{2,2} \\ \vdots \\ x_{2,n} \end{bmatrix}, \ldots, \begin{bmatrix} x_{T,1} \\ x_{T,2} \\ \vdots \\ x_{T,n} \end{bmatrix} \right\}, \tag{1}$$

where $x_{t,i}$ denotes the observation of the ith variable at time t. We assume the existence of at least one seasonal pattern of known length λ within the time series.

Two additional features, $\mathbf{x}_{t,n+1}$ and $\mathbf{x}_{t,n+2}$, are incorporated into each observation $\mathbf{x}_t$ as follows:

$$\mathbf{x}_{t,n+1} = \cos\Theta(t), \qquad \mathbf{x}_{t,n+2} = \sin\Theta(t). \tag{2}$$

The function Θ in Eq. (2) maps the timestamp t to radians:

$$\Theta(t) = t \cdot \frac{2\pi}{\lambda}, \tag{3}$$

where λ represents the length of the seasonal period we intend to be made explicit to the model (e.g., daily or weekly seasonality) expressed in temporal units.

For complex seasonal patterns, our framework can accommodate multiple seasonalities through hierarchical regime modeling. For instance, weekly patterns may exhibit distinct behaviors between weekdays and weekends, or each day may follow a characteristic pattern that repeats over time. The model can explicitly represent these distributional variations by incorporating additional categorical variables that capture the relevant temporal context.

Models within our framework are trained on these $n + 2$ features and required to predict both the original time series features and the temporal coordinate pair $\hat{\mathbf{x}}_{t,n+1}$ and $\hat{\mathbf{x}}_{t,n+2}$, with support for multiple seasonal patterns. The following section presents a model architecture suitable for this purpose.

3.2. Predictions with a conditional variational autoencoder

Let $\mathbf{x}$ denote a time series observation and c a categorical regime variable (e.g., weekday/weekend, day-of-week, holiday status). The forecasting problem can be formulated as estimating the joint distribution of $\mathbf{x}$ conditioned on c. The marginal distribution over observed variables $p_\theta(\mathbf{x}|c)$ is expressed as:

$$p_\theta(\mathbf{x}|c) = \int p_\theta(\mathbf{x}, \mathbf{z}|c)\, d\mathbf{z} = \int p_\theta(\mathbf{z}|c) p_\theta(\mathbf{x}|\mathbf{z}, c)\, d\mathbf{z}, \tag{4}$$

where $\mathbf{z}$ the latent vector.

However, Eq. (4) lacks an efficient analytical solution due to the intractability of the posterior distribution $p_\theta(\mathbf{z}|\mathbf{x}, c)$. To approximate $p_\theta(\mathbf{x}|c)$, we employ a Conditional Variational Autoencoder (cVAE) [35,36], which extends Variational Autoencoders (VAE) [37] to incorporate conditional variables. The cVAE consists of an encoder network that approximates the posterior distribution over latent variables, making posterior inference tractable, and a decoder network that estimates the conditional likelihood.

We propose the architecture shown in Fig. 2, employing Multi-Layer Perceptrons (MLPs) with two fully connected layers for both encoder and decoder networks.

3.2.1. Training

During training, each observation $\mathbf{x}$ is associated with a label c that encodes the seasonal context. For weekly patterns, c indicates the day of the week; for daily patterns, it may represent the hour of the day. This label is not a manual annotation but is deterministically computed from the observation timestamp.

At each training step, batches of observations $\mathbf{x}$ are concatenated with their corresponding labels c and fed to the encoder network. The encoder parametrizes the approximate posterior distribution $q_\theta(\mathbf{z}|\mathbf{x}, c)$ as a diagonal Gaussian with mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$.

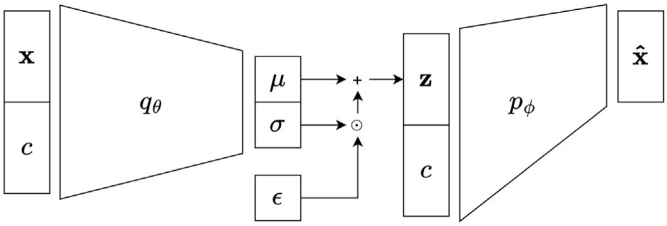


Fig. 2. Conditional variational autoencoder architecture for seasonal time series generation.

To enable gradient-based optimization, we employ the reparametrization trick [37] to sample latent vectors $\mathbf{z}$:

$$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}, \tag{5}$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathbf{I})$ is random noise, and $\odot$ denotes element-wise multiplication. The encoder-derived parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ enable backpropagation while $\boldsymbol{\epsilon}$ maintains the stochastic nature of $\mathbf{z}$.

The decoder network receives the latent vector $\mathbf{z}$ and condition c as input, defining the conditional likelihood $p_\phi(\mathbf{x}|\mathbf{z}, c)$. The decoder output is the reconstructed observation $\hat{\mathbf{x}}$.

The training objective combines reconstruction fidelity with latent space regularization:

$$\min \sum_i \mathcal{L}_{\text{rec}}(\mathbf{x}_i, \hat{\mathbf{x}}_i) + \beta\Big(\mathcal{D}_{\text{KL}}[\mathcal{N}(\mu_i, \sigma_i) \parallel \mathcal{N}(0, I)]\Big), \tag{6}$$

where $\mathcal{L}_{\text{rec}}$ measures reconstruction quality, $\mathcal{D}_{\text{KL}}$ is the Kullback–Leibler divergence ensuring well-structured latent representations, and β controls the regularization strength [38].

3.2.2. Forecasting

After training, the cVAE generates new observations by sampling latent vectors from the prior distribution $\mathbf{z} \sim \mathcal{N}(0, \mathbf{I})$ and transforming them through the decoder conditioned on the desired regime c.

The generative process produces unordered observations distributed along the encoded seasonal pattern of length λ. The timestamp trick [34] enables one to reorder these independent observations using the predicted xy-coordinates of the helix. The encoded features $\hat{\mathbf{x}}_{t,n+1}$ and $\hat{\mathbf{x}}_{t,n+2}$ allow reconstruction of index $\bar{t}$ via:

$$\bar{t} = \text{round}\left(\left(\arctan2\left(\hat{\mathbf{x}}_{t,n+2}, \hat{\mathbf{x}}_{t,n+1}\right) \bmod 2\pi\right) \cdot \frac{\lambda}{2\pi}\right) \bmod \lambda. \tag{7}$$

Modular operations ensure that the output of $\arctan 2$ falls within $[0, 2\pi)$, which is then rounded and assigned to $[0, \lambda)$, establishing ordering relationships among the predictions.

However, the ordered observations form an irregular time series that requires regularization through temporal bucketing for comparison with actual data. Since latent space sampling is stochastic, some temporal buckets may remain empty. We address this by ensuring sufficient sample generation to fill all buckets with high probability.

For a seasonal period discretized into $b = \frac{\lambda}{\nu}$ buckets (where ν is the sampling frequency), we generate m samples such that the probability of filling all buckets is:

$$P(m, b) \approx \left(1 - \Phi\left(-\sqrt{\frac{m(b-1)}{b}}\right)\right)^b \tag{8}$$

where $\Phi(x)$ is the standard normal cumulative distribution function.

This probabilistic framework assumes uniform sampling across buckets and independence between bucket occupancy. The approximation leverages the normal approximation to the binomial distribution for large m.

Once sufficient observations are generated, we perform temporal resampling by aggregating values within each bucket using a suitable function (e.g., mean, median). The number of generated samples m

Table 2
Datasets used in this study.

Dataset	Dataset origin	Dimensions	Size	Anomaly obs.
SynMul16	Synthetic	16	346k	15%
Dodgers Loop Sensors	Real-world	1	4k	6%
Taxi	Real-world	1	10k	10%
Real Tweets (AMZN)	Real-world	1	16k	10%
Real Tweets (FB)	Real-world	1	16k	10%
Real Tweets (GOOG)	Real-world	1	16k	9%
MIT-BIH Arrhythmia	Real-world	14	167k	1%

also controls the prediction variability through the smoothing factor s, where $m = b \cdot s$. Higher values of s increase the likelihood of multiple observations per bucket, resulting in more stable aggregated predictions with reduced intrinsic variability.

3.3. Anomaly detection

During inference, the model predicts at least one complete seasonal period of the expected time series, enabling a real-time comparison between incoming observations and their expected values. The anomaly score for an observation is computed using distance metrics such as Manhattan distance or Euclidean distance, where $\mathbf{x}_t$ represents the actual value and $\hat{\mathbf{x}}_t$ the predicted value:

$$d_1(\mathbf{x}_t, \hat{\mathbf{x}}_t) = \sum_{i=1}^{n} \left| \mathbf{x}_{t,i} - \hat{\mathbf{x}}_{t,i} \right|, \tag{9}$$

$$d_2(\mathbf{x}_t, \hat{\mathbf{x}}_t) = \sqrt{\sum_{i=1}^{n} \left(\mathbf{x}_{t,i} - \hat{\mathbf{x}}_{t,i} \right)^2}. \tag{10}$$

Higher distance values indicate more significant anomalies. Our framework is agnostic to the specific thresholding method employed, accommodating both simple statistical approaches based on distributional properties (such as standardized scores, interquartile ranges, or robust deviation measures) and more sophisticated adaptive techniques. The simpler statistical methods are particularly well-suited for real-time anomaly detection scenarios due to their computational efficiency and minimal memory requirements, making them ideal for deployment in resource-constrained environments where low latency is critical.

4. Experimental setup

We evaluate the RTAD-cVAE model on different datasets using comprehensive baseline comparisons and standardized evaluation metrics. The following subsections delve into the details of the datasets, comparative methods, evaluation metrics, and implementation specifics that constitute our experimental framework and anomaly detection workflow.

4.1. Datasets

Our experiments involved five univariate seasonal time series extracted from public datasets containing real-world data. We also considered a synthetic multivariate seasonal time series to simulate diverse scenarios of anomalies. Additionally, we used a time series without evident seasonality based on a real-world cardiac arrhythmia dataset. Table 2 provides an overview of the datasets used during the experimental phase.

SynMul16 dataset. SynMul16 is a synthetic multivariate time series dataset generated using GutenTAG [39]. The dataset consists of 16 channels derived from fundamental waveform transformations, sampled at 5-s intervals over a 20-day period (345,600 timestamps). The base signal incorporates an electrocardiogram (ECG) pattern with superimposed sinusoidal and cosinusoidal components. The channels exhibit diverse temporal characteristics through systematic transformations of the base signals, including phase-shifted variants (quarter-day and half-day lags), amplitude modulations with daily periodicity, logarithmic and square-root transformations, moving averages, high-frequency residual components, seasonal differencing, and non-linear transformations. This design ensures representation of various signal behaviors commonly encountered in real-world monitoring systems. Anomalies were explicitly injected across multiple channels, comprising approximately 15% of the dataset. These anomalies represent a comprehensive taxonomy of temporal irregularities: mean shifts, amplitude fluctuations, variance changes, platform anomalies (constant values), extremum anomalies (global minima and local maxima), and frequency perturbations.

Dodgers loop sensor dataset. The Dodgers Loop Sensor dataset [40], available from the University of California, Irvine (UCI) machine learning repository, comprises traffic data collected near the Dodgers stadium in Los Angeles. It captures unusual traffic patterns after games, spanning 25 weeks with 288 time slices per day (5 min count aggregates).

NYC Taxi dataset. The NYC Taxi dataset from the Numenta Anomaly Benchmark (NAB) [24] is a univariate time series containing 10,320 observations with 30 min frequency from July 1, 2014, to January 31, 2015. Each observation represents the total number of taxi passengers in NYC at a given time point.

Real Tweets dataset. The Real Tweets dataset from the Numenta Anomaly Benchmark (NAB) [24] comprises Twitter mentions of major publicly traded companies. We selected three representative signals: GOOG, FB, and AMZN. Each signal records the number of mentions for its corresponding symbol every five minutes.

MIT-BIH Arrhythmia dataset. The MIT-BIH Arrhythmia database [41] contains 48 half-hour excerpts of two-channel ambulatory ECG recordings. These recordings have a sampling rate of 360 Hz, an 11-bit resolution, and a range of 10 mV. We selected 14 patients with fewer than 30 annotated anomalies. For each patient, we detected R peaks to extract fixed-length heartbeat windows (144 samples). Each observation was labeled as normal or anomalous based on the original annotations. Subsequently, we constructed a multivariate time series dataset by treating each patient as a separate channel and concatenating the beats. This resulted in a final multivariate time series with 1163 beats.

4.2. Baseline methods

We evaluated our proposed RTAD-cVAE method against established baseline approaches across multiple datasets, conducting comparisons in both semi-supervised and unsupervised settings. The evaluation strategy combined the results of previous studies with our own implementations of established methods to ensure comprehensive coverage.

Our experimental design varied based on available benchmarks. For the SynMul16n dataset, we implemented an unsupervised evaluation against four methods: MAD-AD [4], Prophet [28], COUTA [6], and DIF [5]. We utilized PyOD [42,43] implementations for MAD-AD and DIF, while employing DeepOD [5,6] for COUTA. The dataset was partitioned using a 5:1:4 ratio for training, validation, and test sets. For univariate methods like MAD-AD and Prophet, we trained separate models for each signal.

The Dodgers Loop Sensor dataset evaluation followed a dual approach. First, we compared against results from [31], which implemented OCSVM [2], iForest [3], Piecewise AD [27], LSTM-FD [29], LSTM-AD [30], and AD-LTI [31] in an unsupervised setting. Data preprocessing included hourly frequency aggregation and feature standardization. Additionally, we conducted semi-supervised comparisons against MAD-AD, Prophet, COUTA, and DIF, training on the first 3000 days using only non-game days (1800 observations) and evaluating on the final 1000 days.

For the NYC Taxi dataset, we leveraged two existing studies with semi-supervised approaches. Williams et al. [21] implemented statistical models that included various combinations of Moving Average, SIMA, SARIMA, Multi-SARIMA, and TBATS [22]. We trained our model on three weeks of anomaly-free data and evaluated performance on the remaining data containing five anomalous windows. The second comparison utilized results from [26], which evaluated Numenta [24], NumentaTM [25], Skyline [7], Twitter ADVec [23], and DeepAnT [26] using a 40%–60% development-test split.

The Real Tweets evaluation compared our unsupervised approach against TadGAN [13] and AER [15], two state-of-the-art generative methods. We directly compared the results with the Orion 0.6.0 benchmark [44,45], aggregating the time series into 10 min intervals with z-score normalization.

Finally, for the MIT-BIH Arrhythmia dataset, we conducted semi-supervised comparisons against TranAD, TimesNet, and Anomaly Transformer using DeepOD implementations [5,6]. The first 40% of the dataset served as the training set due to its anomaly-free nature, while the remainder constituted the test set.

4.3. Evaluation criteria

Anomaly detection performance is evaluated using Precision, Recall, F1-score, and Area Under the ROC Curve (AUC-ROC) metrics, as these metrics have been used in previous studies that we are comparing our approach to.

Precision quantifies the model's ability to avoid false alarms by measuring the proportion of correctly identified anomalies among all instances classified as anomalous:

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{11}$$

where TP (True Positives) represents correctly identified anomalies, and FP (False Positives) represents normal instances incorrectly classified as anomalous.

Recall (or sensitivity) measures the model's effectiveness in identifying actual anomalies by calculating the fraction of anomalous instances correctly detected:

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{12}$$

where FN (False Negatives) represents anomalous instances that were incorrectly classified as normal.

The F1-score provides a balanced measure when both false positives and false negatives carry similar importance, combining precision and recall into a single metric:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{13}$$

In time series anomaly detection, precision and recall have been interpreted differently across various studies. To ensure fair comparisons with existing work, we adapted our evaluation methodology for each benchmark by aligning our definitions of TP, FP, and FN with their respective approaches.

For instance, Wu et al. [31] based anomaly scoring on overlapping segments. They defined a true positive (TP) as the detection of at least one anomalous point within an anomalous window, while false positives (FP) were points outside anomalous windows classified as anomalous. In contrast, Sperl et al. [20] considered anomalous windows as single anomalies.

On the other hand, Munir et al. [26] treated values within the outlier window as individual points. Wong et al. [15] counted an observation as a true positive when a known anomalous window overlapped with any detected window. A false negative was when a known anomalous window had no overlap with detected windows, and a false positive was when a detected window did not overlap with any known anomalous region.

We also considered range-based precision (Precision_T) and recall (Recall_T) metrics, introduced by Tatbul et al. [46]. These metrics extend classical precision and recall by considering anomalies as ranges rather than isolated points. They account for partial overlaps between predicted and actual anomaly ranges, providing a more comprehensive evaluation. This approach can be customized to reflect domain-specific priorities regarding detection timing and range fragmentation.

For threshold-independent evaluation, we employed the Area Under the Receiver Operating Characteristic Curve (AUC-ROC), which assesses the model's discriminative ability across various threshold settings:

$$\text{AUC-ROC} = \int_0^1 \text{TPR}(\text{FPR}^{-1}(t))\, dt \tag{14}$$

where TPR represents the true positive rate and FPR the false positive rate. This metric facilitates model comparison and performance assessment across various operating points.

4.4. Implementation details

We developed two variational autoencoder architectures tailored to the complexity of the time series data. For univariate time series, we designed a model with seven conditional categories representing days of the week. The encoder architecture consists of two dense hidden layers with 128 and 64 units respectively, with the decoder employing a symmetric structure. We set the latent space dimensionality to 5 and utilized the Adam optimizer with a learning rate of 0.001. For multivariate time series, we adopted a deeper architecture featuring three hidden layers (256, 128, and 64 units), expanded the latent space to 16 dimensions, and increased the batch size to 64.

The cVAE architecture incorporates inherent regularization via the Kullback–Leibler (KL) divergence term, which constrains the latent space towards the prior distribution. To prevent posterior collapse and promote meaningful representation learning, we apply KL annealing during the initial 30 epochs, progressively increasing the KL weight from 0 to β. Models were trained for 300 epochs with a batch size of 32, using mean squared error as the reconstruction loss. Training stability and generalization were ensured through monitoring of training and validation losses, with early stopping based on reconstruction error on the validation set. The implementation was carried out using TensorFlow 2.0 and Python 3.

Anomaly scores were computed using Eq. (9) across all experiments. We employed dataset-specific thresholding approaches tailored to each dataset's characteristics. For the SynMul16 dataset, we utilized the Peaks-Over-Threshold (POT) technique [47] applied to normalized anomaly scores for each channel. The NYC Taxi and MIT-BIH Arrhythmia datasets employed z-score thresholding [48], while the Real Tweets dataset used signal-specific fixed thresholds (AMZN: 0.7, FB: 0.3, GOOG: 0.7).

Fig. 3 presents an end-to-end overview of the anomaly detection process. During the training phase, the model learns the conditional joint distribution of historical data, with conditions such as day-of-week derived directly from observation timestamps. The preprocessing phase involves data standardization and the addition of engineered features as defined in Eq. (2). During the operational phase, the model generates observations independently by taking latent vectors and temporal conditions as input. The generated observations undergo post-processing

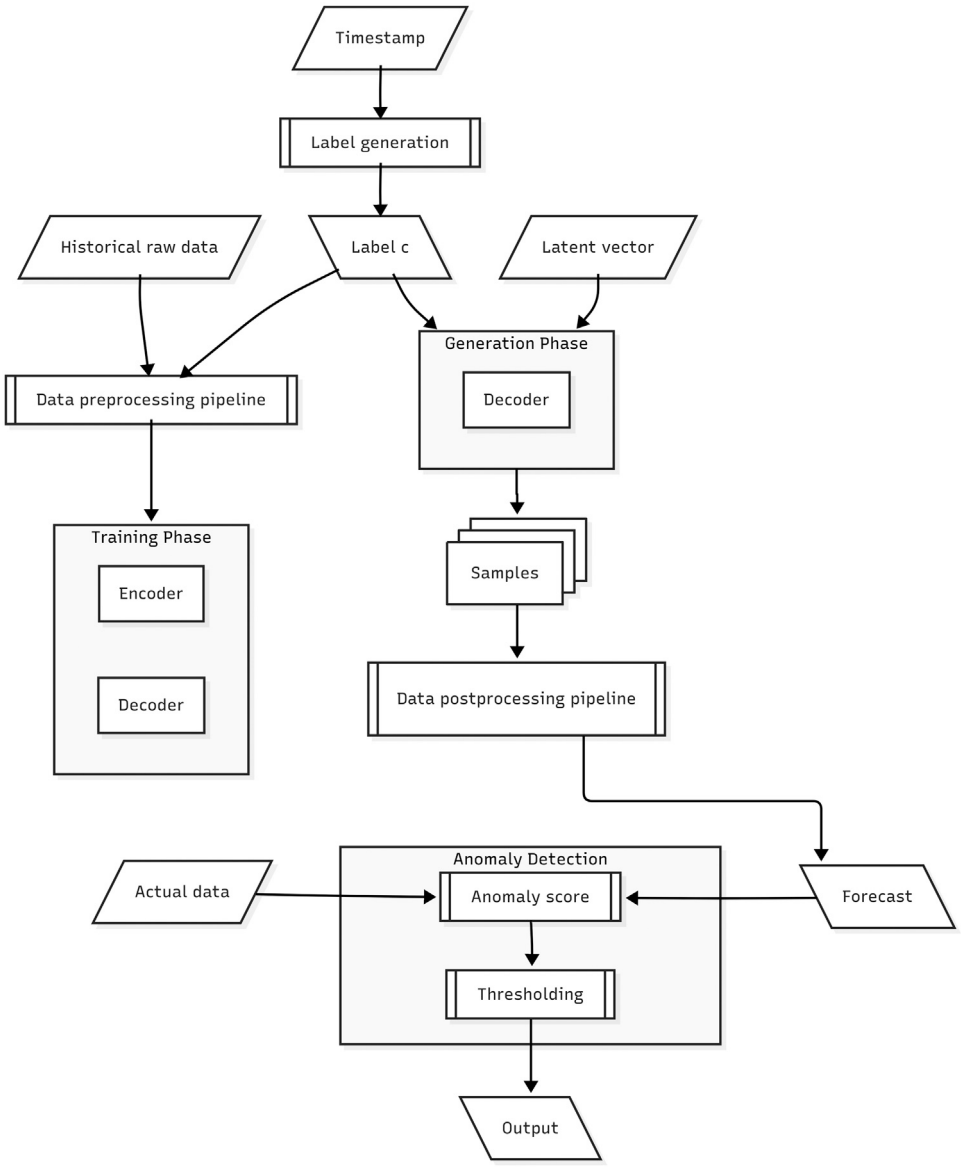


Fig. 3. End-to-end workflow of the proposed anomaly detection system based on RTAD-cVAE. During training, the encoder–decoder learns the conditional distribution of historical data. In the generation phase, the decoder produces samples from latent vectors and conditional labels. The anomaly detection phase compares incoming observations with generated baselines to compute anomaly scores and perform threshold-based classification.

through timestamp reconstruction, enabling proper temporal ordering and aggregation to obtain the expected time series. Finally, each incoming observation is classified as anomalous or normal through threshold-based comparison with the generated baseline.

5. Results

Our comprehensive experimental evaluation of RTAD-cVAE encompasses quantitative comparisons with baseline methods, performance analysis across diverse datasets, ablation studies, thresholding strategy exploration, and investigation of architectural design choices.

5.1. Anomaly detection results

We compare the proposed method against baseline approaches spanning statistical, machine learning, and deep learning paradigms on both synthetic and real-world datasets with diverse seasonal patterns and anomaly characteristics. The semi-supervised and unsupervised experiments were repeated 30 times to account for stochastic variation in the model's generation process, with results reported as mean values accompanied by standard deviations and 95% confidence intervals.

5.1.1. Semi-supervised setting

For the NYC Taxi dataset using window-level evaluation metrics (Table 3), RTAD-cVAE achieved perfect detection with 5 true positives, 0 false positives, and 0 false negatives, resulting in precision, recall, and F1-score of 1.0000. The model demonstrated consistent performance across all 30 experimental runs (standard deviation: ±0.0).

Using point-level evaluation metrics on the NYC Taxi dataset (Table 4), RTAD-cVAE achieved an F1-score of 0.394 (±0.003) with a 95% confidence interval of (0.393, 0.395). Fig. 4 shows the t-SNE (t-distributed Stochastic Neighbor Embedding) projection of the latent space into three dimensions for the NYC Taxi dataset.

For the MIT-BIH Arrhythmia dataset (Table 5), the metric evaluation was conducted on individual observations. On this dataset, RTAD-cVAE achieved an average F1-score of 0.007 (±0.002), with a 95% confidence interval of (0.003, 0.01).

Table 3
NYC Taxi dataset semi-supervised anomaly detection results with window-level anomaly detection metrics. Best values in bold.

Method	TP	FP	FN	Precision	Recall	F1-score
SARIMA[a]	2	1464	3	0.0014	0.4000	0.0027
SIMA[a]	3	1587	2	0.0019	0.6000	0.0038
TBATS[a]	3	1391	2	0.0022	0.6000	0.0043
Multi-SARIMA[a]	4	1425	1	0.0028	0.8000	0.0056
SIMA+SARIMA[a]	3	1072	2	0.0028	0.6000	0.0056
MA[a]	2	654	3	0.0030	0.4000	0.0061
SIMA+Multi-SARIMA[a]	3	475	2	0.0063	0.6000	0.0124
MA+SARIMA[a]	2	131	3	0.0150	0.4000	0.0290
MA+Multi-SARIMA[a]	2	93	3	0.0211	0.4000	0.0400
NumentaTM[a]	4	178	1	0.0220	0.8000	0.0428
RTAD-cVAE	5	0	0	**1.0000**	**1.0000**	**1.0000**

[a] Results from [21]

Table 4
NYC Taxi dataset semi-supervised anomaly detection results with point-level anomaly detection metrics. Best values in bold.

Method	Precision	Recall	F1
Twitter ADVec[a]	0.000	0.000	0.000
Skyline[a]	0.000	0.000	0.000
DeepAnT[a]	**1.000**	0.002	0.004
NumentaTM[a]	0.850	0.006	0.012
Numenta[a]	0.770	0.007	0.014
RTAD-cVAE	0.430	**0.363**	**0.394**

[a] Results from [26]

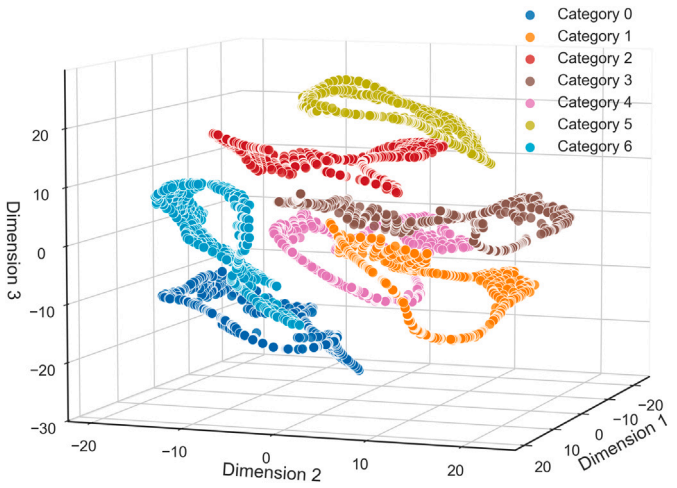


Fig. 4. t-SNE projection of the latent space learned from the NYC Taxi dataset, reduced to three dimensions. Different colors indicate different days of the week (categories 0–6).

Table 5
MIT-BIH Arrhythmia dataset semi-supervised anomaly detection results with point-level anomaly detection metrics. Best values in bold.

Method	Precision	Recall	F1
Anomaly Transformer	0.0028	0.5493	0.0056
TranAD	0.0056	0.8028	0.0112
TimesNet	**0.0090**	0.8451	**0.0179**
RTAD-cVAE	0.0033	**0.9156**	0.0066

5.1.2. Unsupervised setting

On the SynMul16 dataset (Table 6), RTAD-cVAE achieved an F1-score of 0.1422 (±0.0268) with a 95% confidence interval of (0.1311, 0.1532). The model obtained a Precision_T of 0.3992 and Recall_T of 0.0984.

Table 6
SynMul16 dataset unsupervised anomaly detection results. Best values in bold.

Method	Precision_T	Recall_T	F1
Prophet	0.3636	0.0765	0.1264
MAD-AD	0.7941	0.0717	0.1315
DIF	0.9091	0.0714	0.1325
COUTA	**0.9365**	0.0714	0.1328
RTAD-cVAE	0.3992	**0.0984**	**0.1449**

Table 7
Dodgers Loop Sensors dataset unsupervised anomaly detection results. Best value in bold.

Method	AUC-ROC
iForest[a]	0.535
OCSVM[a]	0.591
Piecewise AD[a]	0.751
LSTM-FD[a]	0.829
LSTM-AD[a]	0.859
AD-LTI[a]	0.923
RTAD-cVAE	**0.980**

[a] Results from [31]

For the Dodgers Loop Sensors dataset in the unsupervised setting (Table 7), RTAD-cVAE achieved an AUC-ROC of 0.9796 (±0.0013) with a 95% confidence interval of (0.9791, 0.9801). Fig. 5 illustrates the model's predictions and detected anomalies on this dataset.

On the Real Tweets dataset (Table 8), RTAD-cVAE achieved F1-scores of 0.667, 1.000, and 0.800 for the AMZN, FB, and GOOG signals, respectively.

5.2. Ablation study

Given the intentional simplicity of our model architecture, which employs a basic multi-layer perceptron (MLP), the conditional architecture and smoothing factor represent the primary enhancements distinguishing our approach from a vanilla VAE. We systematically removed each component to quantify its individual contribution to model performance.

For each ablation variant, we trained the model for 30 epochs on the NYC Taxi dataset without early stopping. We evaluated performance using Mean Absolute Error (MAE) on the validation set, which contains only non-anomalous data. MAE was selected as our evaluation metric because it directly measures the model's reconstruction accuracy of normal patterns.

Removing the conditional architecture, which incorporates temporal contextual information (e.g., day of week), resulted in a MAE increase of 0.96% (statistically significant, $p < 10^{-8}$, Cohen's $d = -17.74$).

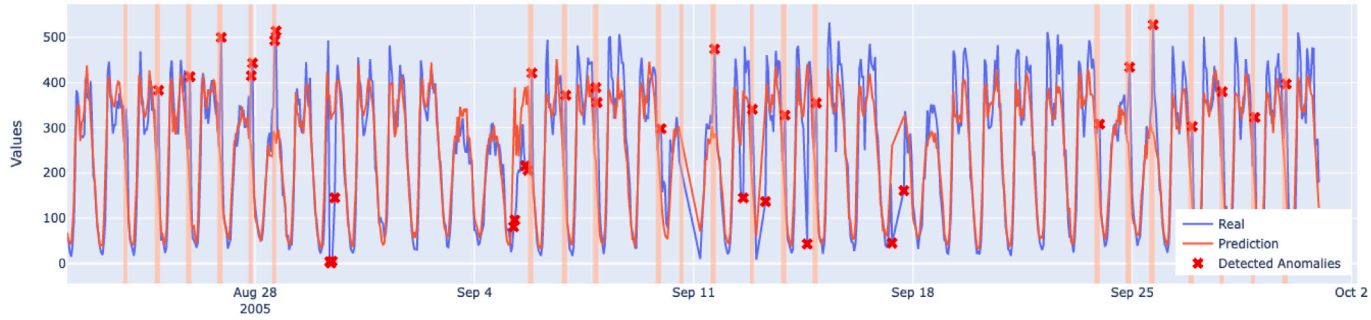


Fig. 5. Anomaly detection results on the Dodgers Loop Sensor dataset. The actual time series is shown in blue, the model's predicted baseline in red, and detected anomalies as red markers. Vertical shaded bands indicate known anomalies.

Table 8
Real Tweets dataset unsupervised anomaly detection results. Best values in bold.

Signal	Method	FP	FN	TP	Recall	Precision	F1
AMZN	TadGAN[a]	0	3	1	0.250	**1.000**	0.400
	AER[a]	1	1	3	0.750	**0.750**	**0.750**
	RTAD-cVAE	0	2	2	0.500	**1.000**	0.667
FB	TadGAN[a]	4	0	2	**1.000**	0.333	0.500
	AER[a]	3	0	2	**1.000**	0.400	0.571
	RTAD-cVAE	0	0	2	**1.000**	**1.000**	**1.000**
GOOG	TadGAN[a]	0	1	2	**0.667**	**1.000**	**0.800**
	AER[a]	0	1	2	**0.667**	**1.000**	**0.800**
	RTAD-cVAE	0	1	2	**0.667**	**1.000**	**0.800**

[a] Results from [45]

Table 9
Baseline hyperparameter configuration for RTAD-cVAE model derived from TPE optimization. These values represent the starting point for sensitivity analysis across different time series datasets.

Hyperparameter	Value
Encoder units	256, 128
Decoder units	128, 256
Latent space size	16
Learning rate	0.0002
KL annealing epochs	15
KL annealing factor	1.0
Smoothing factor s	20

Removing the smoothing factor s caused a substantial 59.32% increase in MAE (statistically significant, $p < 10^{-47}$, Cohen's $d = -53.90$). The smoothing mechanism serves two critical functions: ensuring no empty buckets in predictions (as s is a factor of the parameter m in Eq. (8)) and controlling prediction variability. To handle NaN values during evaluation, we assigned infinity values to null predictions, providing a consistent comparison framework. Fig. 6 illustrates the relationship between smoothing factor values and prediction stability.

All ablation results demonstrate statistical significance with large effect sizes, confirming that both conditioning and smoothing components contribute substantially to model performance.

5.3. Sensitivity analysis

We conducted a sensitivity analysis to examine how key parameters influence model performance, offering practical guidelines for practitioners. We analyzed critical hyperparameters including latent space dimensionality, network architecture, learning rate, and KL annealing settings. Additionally, we investigated data efficiency by evaluating performance with varying training set sizes and assessed inference parameters, including smoothing factor impact and threshold determination methods.

5.3.1. Training hyperparameter sensitivity

We conducted training hyperparameter sensitivity analysis of RTAD-cVAE architecture on SynMul16 and NYC Taxi datasets. We established a baseline model configuration (Table 9) derived through hyperparameter optimization using Tree-structured Parzen Estimator (TPE) [49,50], which navigates the parameter space by modeling distributions of hyperparameters that yield promising results.

The parallel coordinates plots in Figs. 7 and 8 visualize how different hyperparameter combinations affect Mean Absolute Error (MAE), with darker blue lines representing better-performing configurations.

Latent space dimensionality. The model trained on a multivariate dataset achieves the best performance with a latent dimension of 16, while the model trained on a univariate NYC Taxi dataset performs better with dimensions 2 to 4.

Network architecture. A larger model [256,128,64] × [64,128,256] units has proven to perform better on a multivariate time series, while a smaller model [128,64] × [64,128] has achieved better results on a univariate time series.

Learning rate. The model trained on multivariate data performs better with a lower learning rate (approximately 3.89E−04) compared to the model trained on univariate data (approximately 1.05E−05).

KL annealing parameters. Models trained on multivariate datasets benefit from a higher annealing factor, typically over 0.9. In contrast, models trained on univariate datasets perform best with a much lower annealing factor, usually around 0.01.

5.3.2. Training set size sensitivity

We evaluated data efficiency by testing RTAD-cVAE performance with varying portions of the NYC Taxi dataset, ranging from 10% to 50% of available training data. Fig. 9 presents violin plot visualization of reconstruction error (MAE) distributions.

Using 10% of training data yields MAE ≈ 0.37. Increasing to 25% produces substantial improvement (MAE ≈ 0.33, 10.8% reduction). Further increase to 50% achieves MAE ≈ 0.305 (additional 7.6% reduction). The conditional architecture requires exposure to at least one

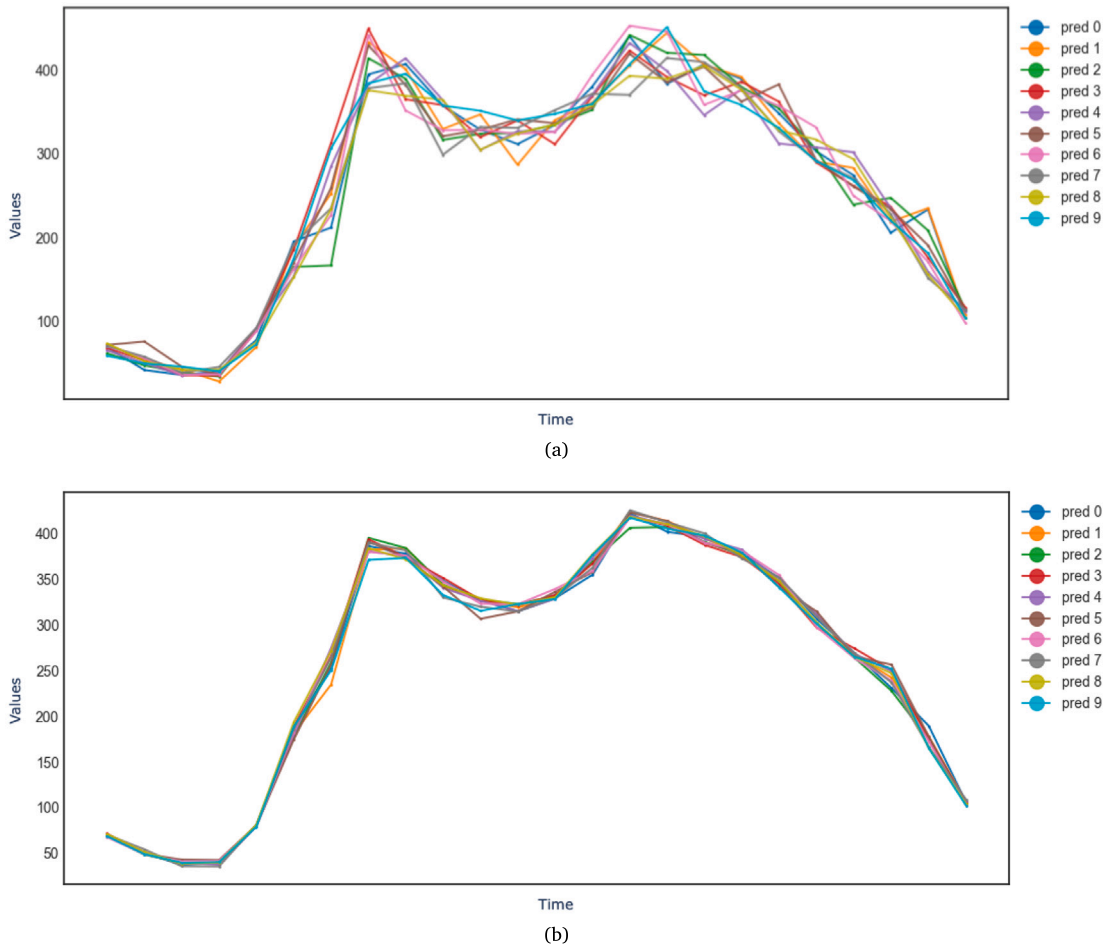


Fig. 6. Ten different generations of the same seasonal period of a time series with different smoothing factors. A higher smoothing factor s reduces forecast variability, resulting in a more stable baseline for anomaly detection. (a) Forecasts with smoothing factor $s = 10$. (b) Forecasts with smoothing factor $s = 100$.

example of each defined conditional category to produce meaningful predictions for that class. The narrow violin plot distributions indicate consistent performance across experimental runs, demonstrating approach stability even with reduced training data.

5.3.3. Inference parameters sensitivity

We analyzed two key inference parameters using the SynMul16 dataset: smoothing factor (s) and anomaly threshold determination methods.

Smoothing factor. The smoothing factor regulates the number of samples drawn from latent space during inference to estimate reconstruction distribution. Fig. 10 shows smoothing factor impact on detection performance and computational requirements. F1 score remains relatively stable across most smoothing values, with modest improvements around $s = 20$ and $s = 140$.

Threshold determination. We evaluated five linear-complexity threshold calculation approaches suitable for high-frequency time series applications:

- **Decomposition-based (DECOMP)** [51]: Applies PCA to decompose cumulative distribution function of reconstruction errors, setting threshold at maximum decomposed matrix value.
- **Z-Score** [48]: Normalizes reconstruction errors, calculating deviation from mean in standard deviation units.
- **Inter-Quartile Region (IQR)** [52]: Sets threshold at $Q_3 + 1.5 \cdot IQR$ where Q_3 is third quartile and IQR is quartile difference.
- **Median Absolute Deviation (MAD)** [53]: Establishes thresholds based on median deviation from median.
- **Peaks-Over-Threshold (POT)** [47]: Based on Extreme Value Theory [54], models error distribution tail using Generalized Pareto Distribution parameters.

Fig. 11 compares methods using F1 scores across multiple experimental runs. Decomposition-based and Z-Score approaches demonstrate superior performance (median F1 scores ≈ 0.21) with low variance. IQR method shows moderate performance (median F1 ≈ 0.15). POT exhibits greater variance, while MAD consistently underperforms (median F1 ≈ 0.125).

5.4. Robustness evaluation

To evaluate the resilience of RTAD-cVAE to real-world data imperfections, we conducted experiments assessing model robustness against two common challenges: measurement noise and missing data. All experiments were performed on the SynMul16 dataset with smoothing factor $s = 20$.

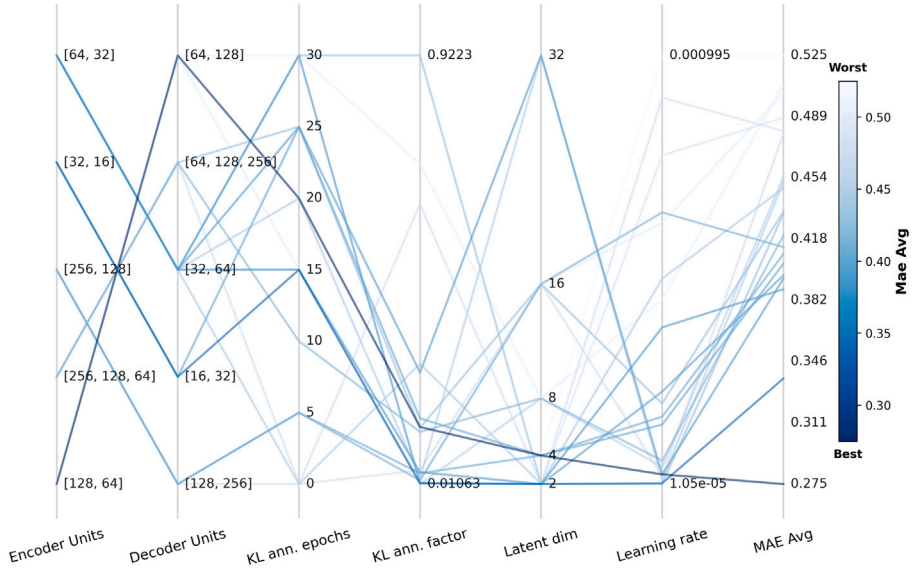


Fig. 7. Parallel coordinates plot of hyperparameter sensitivity analysis on the univariate NYC Taxi dataset. Darker blue lines indicate configurations with lower MAE (better performance).

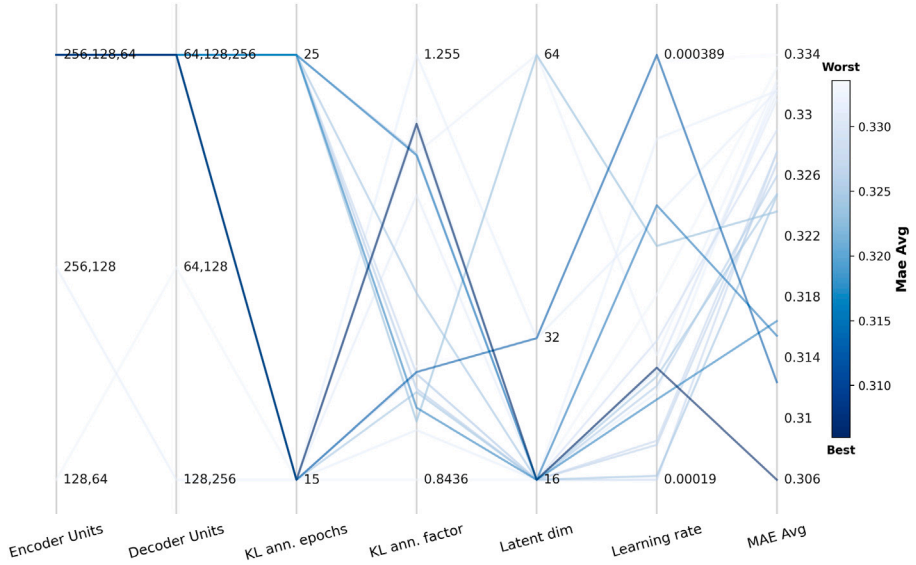


Fig. 8. Parallel coordinates plot of hyperparameter sensitivity analysis on the multivariate synthetic dataset (SynMul16). Darker blue lines indicate configurations with lower MAE (better performance).

Measurement noise. We evaluated the impact of measurement noise by applying Gaussian perturbations to test data at three intensity levels: 10%, 30%, and 50% of the feature standard deviation. Fig. 12 shows model resilience to noise interference. At 10% noise intensity, the model maintains consistent F1 score distribution with median performance ≈0.13 and limited variance. When noise increases to 30%, median performance remains stable, though distribution exhibits greater variability with occasional performance spikes reaching F1 scores of 0.28. At 50% noise level, median F1 score increases slightly to ≈0.14, though with substantially higher variance.

Comparable median performances across noise levels indicate stable core detection capability. The wider performance distribution at higher noise levels demonstrates reduced consistency under severe perturbations.

Missing data. We assessed resilience to data sparsity by randomly removing observations at three levels: 5%, 15%, and 25% of values. Missing values were imputed using feature-wise mean substitution before model training.

Fig. 13 illustrates the detection performance response to increasingly incomplete data. The model maintains relatively stable performance until the 25% threshold, where reliability begins to degrade, particularly evident in the extended lower tail of the distribution and increased variance. Performance remains stable through moderate data loss (5%–15%), with median F1 scores maintaining consistency. At 25% missing values, both median performance and distribution consistency deteriorate, suggesting a critical threshold for reconstruction quality.

The model maintains relatively stable performance until the 25% threshold, where reliability begins to degrade, particularly evident in

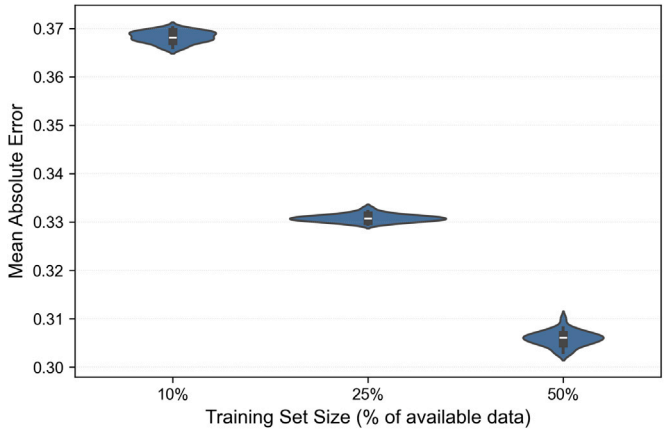


Fig. 9. Impact of training set size on model performance for the NYC Taxi dataset. Violin plots show the distribution of Mean Absolute Error (MAE) across multiple experimental runs as the percentage of available training data increases from 10% to 50%. Lower MAE values indicate better reconstruction performance. The results demonstrate significant performance improvements when increasing from 10% to 25% of training data, with more modest gains when further increasing to 50%.

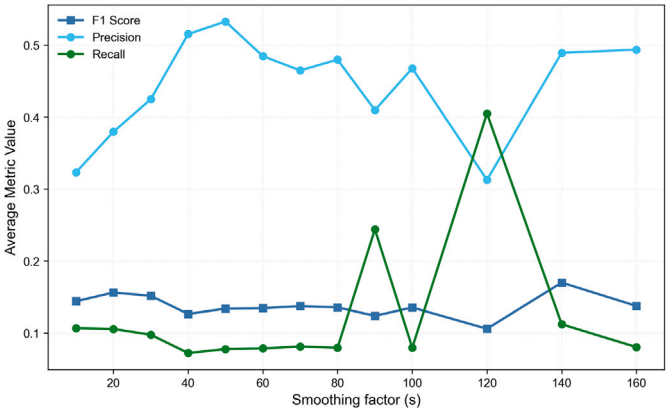


Fig. 10. Impact of smoothing factor s on model performance metrics. F1 score remains relatively stable across most smoothing values, with modest improvements around $s = 20$ and $s = 140$.

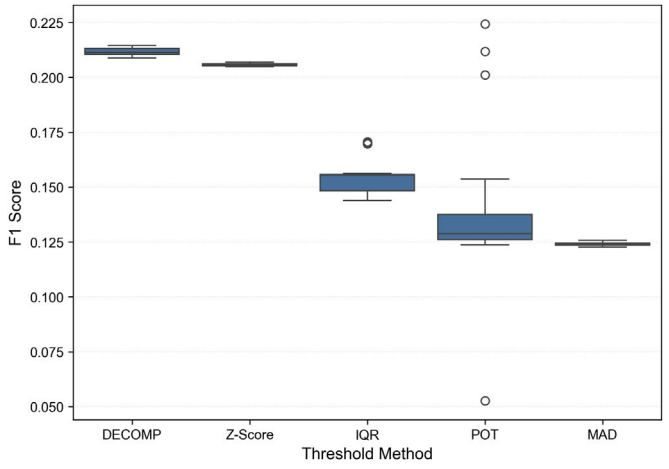


Fig. 11. F1 score distribution comparison across different threshold determination methods.

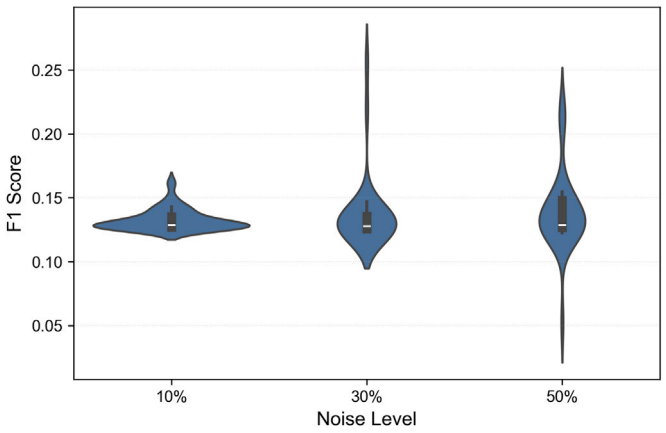


Fig. 12. Impact of measurement noise on anomaly detection performance. Box plots show F1 score distributions across noise intensities of 10%, 30%, and 50% of feature standard deviation.

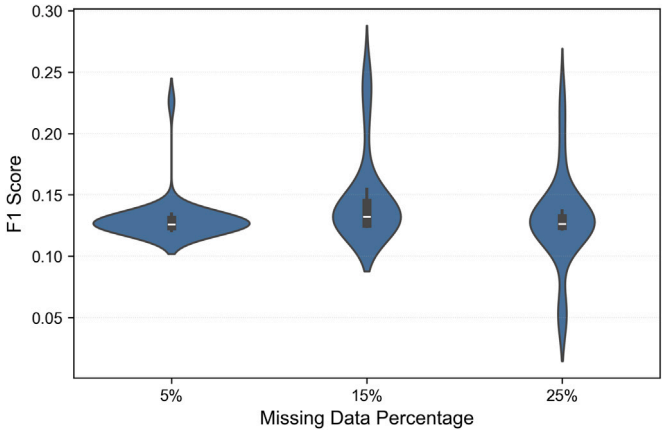


Fig. 13. Model resilience to missing data across different sparsity levels. Box plots show F1 score distributions for 5%, 15%, and 25% missing values.

the extended lower tail of the distribution. This suggests that while RTAD-cVAE exhibits reasonable tolerance to moderate data loss, there exists a critical threshold beyond which reconstruction quality and anomaly detection capability become increasingly unpredictable.

These experiments demonstrate that RTAD-cVAE maintains functional detection capability under challenging data conditions, with performance degradation occurring gradually rather than catastrophically. The conditional architecture likely contributes to this robustness by leveraging temporal context to compensate for local data imperfections. These characteristics make the approach suitable for industrial applications where sensor noise and occasional missing measurements are common operational challenges.

5.5. Scalability

We conducted a comprehensive scalability analysis to evaluate the computational characteristics of our model across varying data dimensionality, seasonal period length λ, and smoothing factor. All experiments employed the SynMul16 dataset as the base configuration, with systematic modifications to assess scalability properties. The experiments were carried out on a MacBook Air (2020) with an Apple M1 chip and 8 GB RAM to evaluate the deployment characteristics in resource-constrained environments.

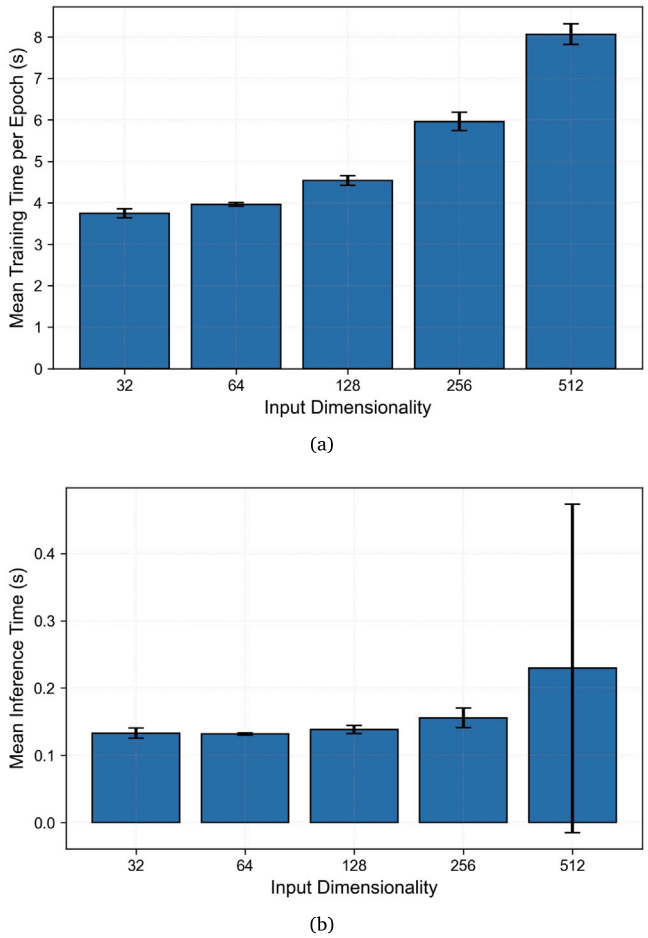


Fig. 14. Impact of data dimensionality on processing times for a seasonal period of one day. (a) Training time versus data dimensionality. (b) Inference time versus data dimensionality.

Data dimensionality. We evaluated computational scalability across varying input dimensionalities by systematically replicating channels from the original 16-dimensional SynMul16 dataset to achieve target dimensionalities of 32, 64, 128, 256, and 512 dimensions. This replication approach provided controlled scaling of dimensionality. The seasonal period λ was fixed at 17,280 timestamps (corresponding to one day at 5-s sampling intervals), and no smoothing factor was applied ($s = 1$). The dataset length was maintained at 345,600 timestamps (20 days) for training consistency across all dimensionality experiments.

Fig. 14 shows the relationship between input dimensionality and both training time and inference time. Training time increases progressively from approximately 3.8 s at 32 dimensions to 8.1 s at 512 dimensions. Inference time remains relatively stable across dimensionalities, ranging from 0.13 to 0.23 s, with a notable increase in variance at 512 dimensions.

Length of the seasonal period λ. We assessed computational requirements across different seasonal period lengths λ by testing values of 4320 (6 h), 8640 (12 h), 17,280 (1 day), and 120,960 (1 week) timestamps at the original 5-s sampling frequency. The dimensionality was maintained at the original 16 channels, and no smoothing factor was applied ($s = 1$).

The seasonal period length affects only the inference phase, as it modifies only the value of the additional variables used for the helical encoding while keeping the training dataset size constant. During inference, longer seasonal periods require generating more samples to fill the entire prediction window, thus increasing the number of generations proportionally to the period length.

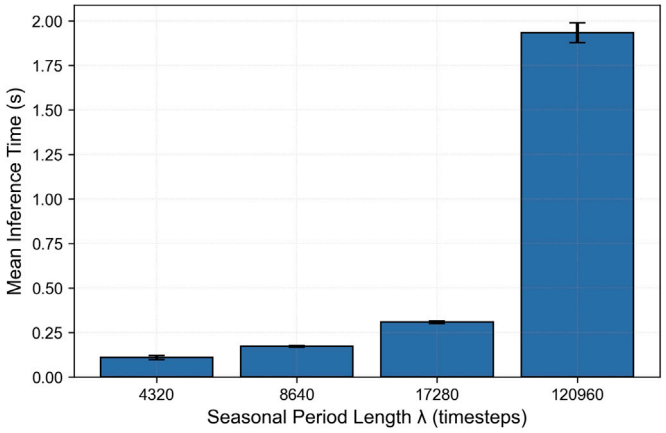


Fig. 15. Impact of the seasonal period λ on the inference time.

Fig. 15 demonstrates the impact of seasonal period λ on inference time. Inference time increases from approximately 0.1 s for 6-h periods to 1.95 s for weekly periods, showing a near-linear relationship with seasonal period length.

Smoothing factor. We analyzed the computational impact of varying smoothing factors s from 20 to 160 using the original SynMul16 dataset configuration (16 dimensions, $\lambda = 17{,}280$). Figs. 16(a) and 16(b) demonstrate the processing time implications. Post-processing time exhibits linear growth from approximately 0.2 s at $s = 20$ to 1.6 s at $s = 160$. Prediction time increases exponentially, rising from approximately 1 s at $s = 20$ to 12–14 s at $s = 160$. The sharp increases beyond $s = 120$ indicate potential memory pressure as the number of latent samples approaches 2.7 million for the one-day seasonal pattern configuration.

5.6. Anomaly detection in streaming data

To evaluate computational efficiency of RTAD-cVAE in real-world conditions, we designed a controlled streaming simulation experiment comparing our approach against an Autoregressive Long Short-Term Memory (AR-LSTM), selected as a representative prediction-based deep learning method. All experiments were conducted on a MacBook Air (2020) with Apple M1 chip and 8 GB RAM.

We simulated continuous operation with predictions executed at 5-s intervals, approximating deployment conditions for high-frequency time series. Resource monitoring was performed at 500 ms intervals, tracking CPU utilization, memory consumption, and energy usage. Carbon emissions were measured via the CodeCarbon library [55].

Fig. 17 presents a comparison of cumulative energy consumption between models. The fundamental architectural difference produces substantially different efficiency patterns. AR-LSTM exhibits linear growth in energy consumption due to the sliding window approach (requiring predictions every 5 s), while RTAD-cVAE generates forecasts for entire seasonal periods in single forward passes.

After 15 min of operation, AR-LSTM consumed 0.00108 kWh compared to 0.00016 kWh for RTAD-cVAE. Projected daily consumption shows 0.1037 kWh versus $8.257 \cdot 10^{-5}$ kWh respectively, representing a 1255-fold efficiency improvement.

Fig. 18 reveals corresponding differences in carbon emissions. AR-LSTM's recurring predictions generate linearly increasing CO_2 emissions, reaching 0.00036 kg after 15 min and projecting to 0.0343 kg for full day operation. RTAD-cVAE's one-shot approach results in $2.731 \cdot 10^{-5}$ kg of CO_2 for equivalent temporal coverage.

Fig. 19 illustrates operational differences between approaches. RTAD-cVAE exhibits a single significant CPU spike during initial inference, followed by minimal computational activity for anomaly detection. AR-LSTM shows recurrent CPU utilization spikes every 5 s,

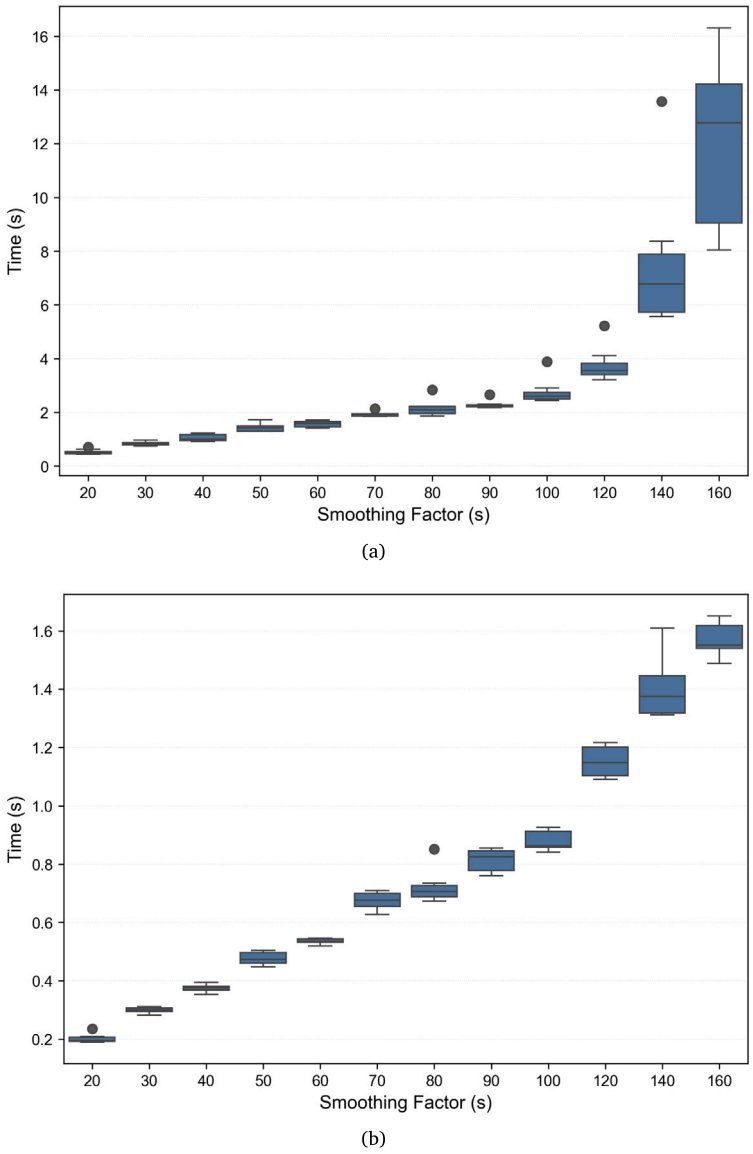


Fig. 16. Impact of smoothing factor s on processing times for a seasonal period of one day. (a) Prediction time versus smoothing factor. (b) Postprocessing time versus smoothing factor.

creating sustained resource pressure. Memory utilization reveals that while RTAD-cVAE initially consumes more memory, it steadily decreases as operations complete. AR-LSTM maintains relatively consistent memory usage between 2–3.5% throughout monitoring period due to continuous sliding window operations.

6. Discussion

Results demonstrate RTAD-cVAE's effectiveness for real-time anomaly detection in seasonal time series data, with particular operational advantages in terms of performance and air-gapped network deployment compared to prediction-based anomaly detection methods. This section discusses the key findings, implications, and limitations of our approach.

6.1. Anomaly detection performance

Across all seasonal time series, RTAD-cVAE demonstrated detection performance comparable to state-of-the-art methods in both semi-supervised and unsupervised settings. In semi-supervised scenarios on the NYC Taxi dataset, the method outperformed all baselines regardless of window-level or point-level evaluation. On the Dodgers Loop Sensors dataset, the model achieved the highest AUC-ROC score.

In unsupervised settings, RTAD-cVAE matched or exceeded existing techniques depending on dataset characteristics. On SynMul16, it yielded the highest F1-score, despite lower precision compared to threshold-based baselines. On the Dodgers dataset, it obtained the best AUC-ROC, surpassing both statistical and deep learning-based methods. On the Real Tweets dataset, RTAD-cVAE results aligned with two state-of-the-art generative methods.

However, performance degrades on non-seasonal time series. Without identifiable patterns to encode and utilize for prediction, the method achieved results below other state-of-the-art approaches, as demonstrated on the MIT-BIH Arrhythmia database. This limitation reflects the framework's design specificity for seasonal time series, leveraging the helical encoding and the timestamp trick for baseline generation.

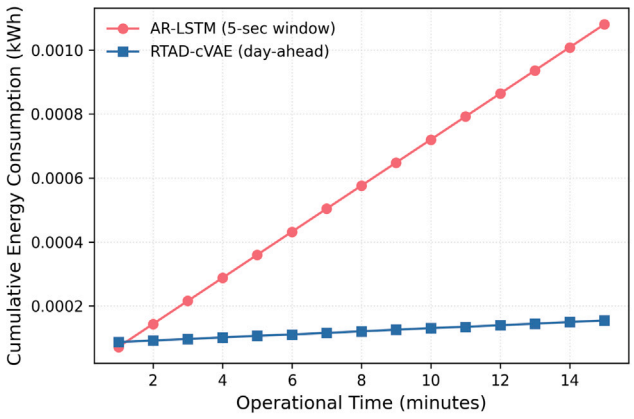


Fig. 17. Cumulative energy consumption of RTAD-cVAE (one-shot day-ahead) versus AR-LSTM (5-s sliding window) over 15 min of continuous operation. The graph demonstrates AR-LSTM's linear growth in energy requirements due to recurring predictions, while RTAD-cVAE's energy consumption remains nearly constant after initial inference.

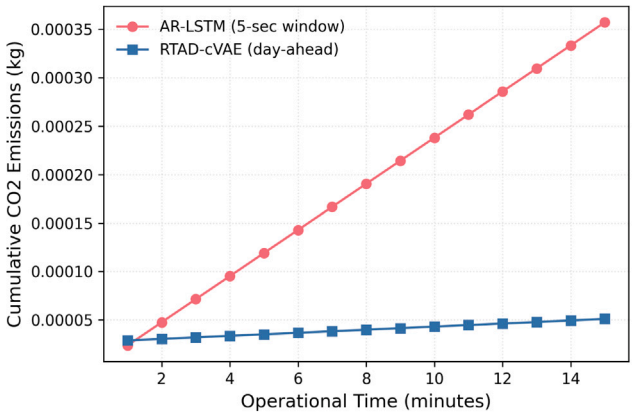


Fig. 18. Cumulative CO_2 emissions from RTAD-cVAE and AR-LSTM during 15 min of continuous operation. The substantial difference in environmental impact results from RTAD-cVAE's ability to generate complete day-ahead forecasts in a single inference step versus AR-LSTM's requirement for repeated sliding window predictions.

6.2. Computational efficiency and scalability

Since the model does not require historical data buffering, it exhibits significant resource usage advantages. Computational load occurs as a single burst during seasonal period prediction, which is typically larger than sliding window sizes but eliminates continuous processing overhead. This advantage becomes more pronounced in less dynamic contexts, making RTAD-cVAE particularly suitable for applications requiring extended temporal forecasting horizons.

The scalability analysis reveals favorable computational characteristics for practical deployment on commodity hardware. Training time scales sub-linearly with dimensionality, while inference time remains largely independent of input dimensions. The linear relationship between seasonal period length and inference time provides predictable computational costs for different temporal scales.

The primary computational bottleneck emerges from the smoothing factor s. While prediction is performed in advance for multiple periods without dependency on real-time data arrival, large smoothing factors in high-dimensional time series increase memory pressure. This can cause performance degradation when resources are constrained, though production-level code optimizations can mitigate this issue.

6.3. Energy efficiency and environmental impact

The windowless prediction capability eliminates computational overhead associated with maintaining and updating sliding windows, resulting in substantial energy advantages. Resource monitoring shows that RTAD-cVAE exhibits a single significant CPU spike during initial long-term forecasting, followed by minimal computational activity. After generating predictions, anomaly detection requires only distance calculations rather than continuous neural network inference.

Quantitative measurements demonstrated a 1255-fold reduction in energy consumption compared to AR-LSTM sliding window approaches, with daily consumption of just $8.257 \cdot 10^{-5}$ kWh versus 0.1037 kWh for sequential prediction-based methods. Environmental impact assessment showed a corresponding 1255-fold reduction in CO_2 emissions—just $2.731 \cdot 10^{-5}$ kg for full-day operation versus 0.0343 kg for sliding window approaches.

6.4. Model robustness and adaptability

Model robustness evaluation revealed RTAD-cVAE's resilience to common data imperfections. The model maintained consistent median F1 scores across increasing noise levels, demonstrating that latent representations capture fundamental data patterns rather than superficial features. Performance remained stable with up to 15% randomly missing observations, with reliability degrading only at the 25% threshold.

The hyperparameter sensitivity analysis reveals fundamental differences between univariate and multivariate processing requirements. Univariate data benefits from simpler architectures with lower regularization, while multivariate data requires higher-capacity models with stronger regularization to capture interdependencies effectively. The smoothing factor serves dual purposes: ensuring mathematical stability by preventing empty prediction buckets and controlling prediction variability to establish reliable baselines for anomaly detection.

6.5. Operational advantages

Beyond performance benefits, RTAD-cVAE delivers substantial operational advantages through its one-shot prediction approach. The modest resource requirements enable deployment on edge devices, facilitating anomaly detection directly at data sources without requiring constant communication with centralized servers. By not storing or requiring access to previous real-time data streams, the approach limits the attack surface for data breaches, enhancing privacy and security.

The windowless prediction capability provides several significant advantages over sequential prediction-based approaches: elimination of window size selection parameters, resolution of the latency-accuracy tradeoff inherent in sliding window methods, and reduced error propagation compared to autoregressive long-term prediction approaches. The timestamp encoding enables both accurate prediction and zero-latency detection by decoupling context requirements from detection latency.

6.6. Explainability considerations

While RTAD-cVAE's latent representations lack direct physical interpretation, the model provides meaningful explainability at the decision level. By generating complete seasonal baselines that can be directly compared with incoming observations, it produces transparent anomaly scores derived from observable differences between real and generated data. The generated baseline serves as a visual explanation tool, allowing users to immediately understand why specific points were flagged by observing their deviation from expected patterns.

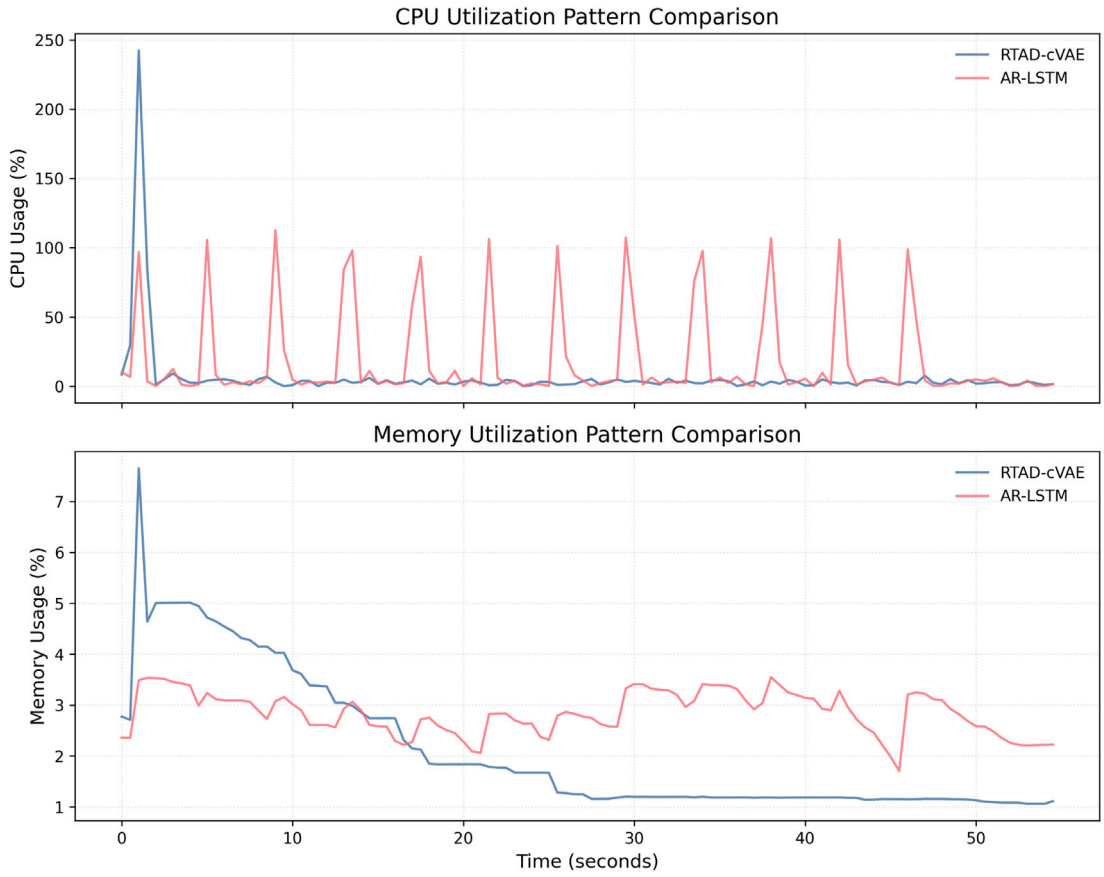


Fig. 19. Resource utilization comparison between RTAD-cVAE and AR-LSTM during one minute of streaming inference on the SynMul16 high-frequency multivariate time series dataset. The top panel shows CPU utilization patterns, with RTAD-cVAE exhibiting a single processing spike followed by minimal activity, while AR-LSTM demonstrates recurring computational demands. The bottom panel displays memory usage over time, showing how RTAD-cVAE releases most allocated memory immediately after initial forecast, resulting in steadily declining footprint, whereas AR-LSTM maintains sustained memory utilization throughout the inference period.

6.7. Limitations and future work

The proposed anomaly detection framework is specifically designed for seasonal time series. This represents a design choice rather than a technical shortcoming, as the framework's advantages emerge specifically from seasonal pattern exploitation for training lightweight generative models based on multilayer perceptrons.

RTAD-cVAE is designed to detect anomalies representing deviations from expected seasonal patterns learned during training. The model operates on representations derived from the training data and does not incorporate real-time adaptive mechanisms or online learning capabilities. Consequently, concept drift — where the underlying data distribution changes — presents a significant challenge. Structural breaks in time series that cause permanent shifts or changes to seasonal patterns may invalidate the model's learned representations. This limitation currently necessitates periodic retraining on updated data after concept drift has been identified, since the model cannot autonomously adapt to distributional shifts without retraining intervention. Future research should develop adaptive mechanisms that enable the framework to automatically adjust to distributional changes without requiring complete retraining, thereby providing more robust long-term deployment capabilities in evolving contexts.

Although the model can automatically learn multiple seasonal patterns simultaneously (e.g., weekly and intra-day patterns as demonstrated with the NYC Taxi demand dataset), the current framework's applicability to multivariate time series is limited to cases where signals share at least one common seasonal pattern. This constraint arises from the need to provide an explicit value for the seasonal period parameter λ for helix encoding representation, which is subsequently utilized for calculating the indexes that allow reordering of the generated observations. When common seasonality across dimensions is not present, superior results would be obtained by training individual models for each seasonal signal within the multivariate series. Future research should address enhancing multi-seasonal pattern handling to allow simultaneous capture of hierarchical temporal patterns across multiple timescales in multivariate series without assuming uniform seasonality across dimensions.

The current version of RTAD-cVAE presents a partial implementation of the helical encoding, utilizing only the x and y coordinates that are exploited by the timestamp mechanism for reconstruction. A complete implementation incorporating the z-axis of the helix as an additional conditioning mechanism could enable trend learning capabilities, potentially enhancing the model's representational capacity. Such an extension would implement native trend modeling and eliminate the current dependency on external trend preprocessing by developing integrated mechanisms within the architecture.

The method currently requires hyperparameter tuning adapted to the complexity of each time series, as indicated by the hyperparameter analysis showing dataset-specific configuration requirements. The selection of appropriate seasonal period length λ for the helical encoding and conditional labels currently requires domain expertise, and these parameters can significantly impact model classification quality. Agent-based automated hyperparameter optimization guided by dataset characteristics and application context would make the method accessible to practitioners without deep domain expertise, reducing the manual tuning burden currently required for optimal performance.

7. Conclusions

We introduced a novel generative framework for real-time anomaly detection in seasonal time series that fundamentally reimagines the prediction-based anomaly detection paradigm by eliminating the dependency on historical data buffering during inference. Through a conditional Variational Autoencoder architecture enhanced with helical encoding and timestamp conditioning mechanisms, the proposed RTAD-cVAE model generates long-term seasonal baselines without requiring historical data input at inference time.

The experimental evaluation across diverse datasets demonstrates that RTAD-cVAE achieves state-of-the-art performance while delivering superior computational efficiency. This paradigm shift enables efficient anomaly detection that maintains competitive accuracy while dramatically reducing computational overhead. These characteristics position RTAD-cVAE as particularly valuable for large-scale deployments, IoT applications, and scenarios where environmental impact and resource efficiency are paramount concerns.

The primary limitation of the proposed method is its applicability exclusively to seasonal time series. While we recognize the value of generalizable approaches, we chose to prioritize performance excellence in this specific, yet important, domain rather than developing a one-size-fits-all solution. Future work could explore adaptations for trending data, though non-seasonal time series would require fundamentally different techniques than the proposed framework based on timestamp conditioning and helical encoding mechanisms.

CRediT authorship contribution statement

Lorenzo Porcelli: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Investigation, Data curation, Conceptualization, Methodology. **Marcello Trovati:** Writing – review & editing, Supervision. **Francesco Palmieri:** Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was partially supported by project SERICS (PE00000014) under the NRRP MUR program funded by the EU - NGEU.

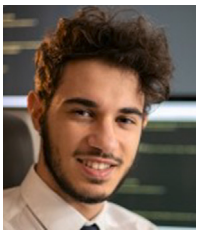
Data availability

The data and code supporting the findings of this study are available at https://bit.ly/3Jsh3dv.

References

[1] V. Chandola, A. Banerjee, V. Kumar, Anomaly detection: A survey, ACM Comput. Surv. 41 (3) (2009) 1–58.
[2] B. Schölkopf, R.C. Williamson, A. Smola, J. Shawe-Taylor, J. Platt, Support vector method for novelty detection, Adv. Neural Inf. Process. Syst. 12 (1999).
[3] F.T. Liu, K.M. Ting, Z.-H. Zhou, Isolation-based anomaly detection, ACM Trans. Knowl. Discov. from Data (TKDD) 6 (1) (2012) 1–39.
[4] B. Iglewicz, D. Hoaglin, The ASQC basic references in quality control: statistical techniques, How To Detect. Handle Outliers. 16 (1993) 1–87.
[5] H. Xu, G. Pang, Y. Wang, Y. Wang, Deep isolation forest for anomaly detection, IEEE Trans. Knowl. Data Eng. 35 (2023) 12591–12604.
[6] H. Xu, Y. Wang, S. Jian, Q. Liao, Y. Wang, G. Pang, Calibrated one-class classification for unsupervised time series anomaly detection, IEEE Trans. Knowl. Data Eng. (2024).
[7] A. Stanway, Skyline, 2013, https://github.com/etsy/skyline (accessed: 10 July 2024).
[8] H. Ringberg, A. Soule, J. Rexford, C. Diot, Sensitivity of pca for traffic anomaly detection, in: Proceedings of the 2007 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, 2007, pp. 109–120.
[9] R.-J. Hsieh, J. Chou, C.-H. Ho, Unsupervised online anomaly detection on multivariate sensing time series data for smart manufacturing, in: 2019 IEEE 12th Conference on Service-Oriented Computing and Applications, SOCA, IEEE, 2019, pp. 90–97.
[10] D.P. Kingma, M. Welling, et al., An introduction to variational autoencoders, Found. Trends® Mach. Learn. 12 (4) (2019) 307–392.
[11] D. Park, Y. Hoshi, C.C. Kemp, A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder, IEEE Robot. Autom. Lett. 3 (3) (2018) 1544–1551.
[12] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, S.-K, Ng, mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks, in: International Conference on Artificial Neural Networks, Springer, 2019, pp. 703–716.
[13] A. Geiger, D. Liu, S. Alnegheimish, A. Cuesta-Infante, K. Veeramachaneni, 2020 Ieee International Conference on Big Data (Big Data), in: Tadgan: Time series anomaly detection using generative adversarial networks, IEEE, 2020, pp. 33–43.
[14] B. Zhou, S. Liu, B. Hooi, X. Cheng, J. Ye, Beatgan: Anomalous rhythm detection using adversarially generated time series., IJCAI 2019 (2019) 4433–4439.
[15] L. Wong, D. Liu, L. Berti-Equille, S. Alnegheimish, K. Veeramachaneni, Aer: Auto-encoder with regression for time series anomaly detection, in: 2022 IEEE International Conference on Big Data, Big Data, IEEE, 2022, pp. 1152–1161.
[16] J. Xu, H. Wu, J. Wang, M. Long, Anomaly transformer: Time series anomaly detection with association discrepancy, 2021, ArXiv Preprint arXiv:2110.02642.
[17] S. Tuli, G. Casale, N. Jennings, Tranad: deep transformer networks for anomaly detection in multivariate time series data, Proc. the VLDB Endow. 15 (2022) 1201–1214, http://dx.doi.org/10.14778/3514061.3514067.
[18] H. Wu, T. Hu, Y. Liu, H. Zhou, J. Wang, M. Long, Timesnet: Temporal 2d-variation modeling for general time series analysis, 2022, arXiv Preprint arXiv:2210.02186.
[19] R. Hyndman, G. Athanasopoulos, Forecasting: Principles and Practice, OTexts, 2018.
[20] R.E. Sperl, S.M. Chung, Two-step anomaly detection for time series data, in: 2019 International Conference on Data and Software Engineering, ICoDSE, IEEE, 2019, pp. 1–5.
[21] A. Williams, R. Sperl, S. Chung, Anomaly detection in multi-seasonal time series data, IEEE Access. 11 (2023) 106456–106464.
[22] A. De Livera, R. Hyndman, R. Snyder, Forecasting time series with complex seasonal patterns using exponential smoothing, J. Am. Stat. Assoc. 106 (2011) 1513–1527.
[23] Twitter, Anomaly detection r package, 2015, https://github.com/twitter/AnomalyDetection/releases (Accessed: 10 July 2024).
[24] A. Lavin, S. Ahmad, Evaluating real-time anomaly detection algorithms–the numenta anomaly benchmark, in: 2015 IEEE 14th International Conference on Machine Learning and Applications, ICMLA, IEEE, 2015, pp. 38–44.
[25] S. Ahmad, A. Lavin, S. Purdy, Z. Agha, Unsupervised real-time anomaly detection for streaming data, Neurocomputing 262 (2017) 134–147.
[26] M. Munir, S.A. Siddiqui, A. Dengel, S. Ahmed, Deepant: A deep learning approach for unsupervised anomaly detection in time series, Ieee Access 7 (2018) 1991–2005.
[27] O. Vallis, J. Hochenbaum, A. Kejariwal, A novel technique for {long-term} anomaly detection in the cloud, in: 6th USENIX Workshop on Hot Topics in Cloud Computing, HotCloud 14, 2014.
[28] S. Taylor, B. Letham, Forecasting at scale, Am. Stat. 72 (2018) 37–45.
[29] P. Filonov, A. Lavrentyev, A. Vorontsov, Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model, 2016, arXiv preprint arXiv:1612.06676.
[30] P. Malhotra, L. Vig, G. Shroff, P. Agarwal, et al., Long short term memory networks for anomaly detection in time series, in: Esann, Vol. 2015, 2015, p. 89.
[31] W. Wu, L. He, W. Lin, Y. Su, Y. Cui, C. Maple, S. Jarvis, Developing an unsupervised real-time anomaly detection scheme for time series with multi-seasonality, IEEE Trans. Knowl. Data Eng. 34 (9) (2022) 4147–4160.
[32] E.H. Pena, M.V. de Assis, M.L. Proença, Anomaly detection using forecasting methods arima and hwds, in: 2013 32nd International Conference of the Chilean Computer Science Society, sccc, IEEE, 2013, pp. 63–66.
[33] K. Hundman, V. Constantinou, C. Laporte, I. Colwell, T. Soderstrom, Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 387–395.
[34] L. Porcelli, U. Fiore, F. Palmieri, Generative models with helical time encoding for seasonal time series forecasting, Eng. Appl. Artif. Intell. (2024) (submitted for publication).
[35] D.P. Kingma, S. Mohamed, D. Jimenez Rezende, M. Welling, Semi-supervised learning with deep generative models, Adv. Neural Inf. Process. Syst. 27 (2014).
[36] K. Sohn, H. Lee, X. Yan, Learning structured output representation using deep conditional generative models, Adv. Neural Inf. Process. Syst. 28 (2015).

[37] D. Kingma, M. Welling, Auto-encoding variational bayes, in: 2nd International Conference on Learning Representations, iclr2014, 2014, Preprint, submitted for publication.
[38] S.R. Bowman, L. Vilnis, O. Vinyals, A.M. Dai, R. Jozefowicz, S. Bengio, Generating sentences from a continuous space, 2015, arXiv preprint arXiv:1511.06349.
[39] P. Wenig, S. Schmidl, T. Papenbrock, TimeEval: A benchmarking toolkit for time series anomaly detection algorithms, Proc. the VLDB Endow. (PVLDB). 15 (2022) 3678–3681.
[40] J. Hutchins, Dodgers loop sensor, UCI Mach. Learn. Repos. (2006) http://dx.doi.org/10.24432/C51P50.
[41] MIT Laboratory for Computational Physiology, MIT-bih arrhythmia database, version 1.0.0, PhysioNet, 2005, http://dx.doi.org/10.13026/C2F305, published Available at:.
[42] Y. Zhao, Z. Nasrullah, Z. Li, Pyod: A python toolbox for scalable outlier detection, J. Mach. Learn. Res. 20 (2019) 1–7, http://jmlr.org/papers/v20/19-011.html.
[43] S. Chen, Z. Qian, W. Siu, X. Hu, J. Li, S. Li, Y. Qin, T. Yang, Z. Xiao, W. Ye, Y. Zhang, Y. Dong, Y. Zhao, Pyod 2: A python library for outlier detection with LLM-powered model selection, 2024, ArXiv Preprint arXiv:2412.12154.
[44] Orion benchmark 0.6.0, 2022, https://github.com/sintel-dev/Orion, (Accessed 10 July 2024).
[45] S. Alnegheimish, D. Liu, C. Sala, L. Berti-Equille, K. Veeramachaneni, Sintel: A machine learning framework to extract insights from signals, in: Proceedings of the 2022 International Conference on Management of Data, 2022, pp. 1855–1865.
[46] N. Tatbul, T. Lee, S. Zdonik, M. Alam, J. Gottschlich, Precision and recall for time series, Adv. Neural Inf. Process. Syst. 31 (2018).
[47] A. Siffer, P. Fouque, A. Termier, C. Largouet, Anomaly detection in streams with extreme value theory, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 1067–1075.
[48] V. Bagdonavičius, L. Petkevičius, Multiple outlier detection tests for parametric models, Math. 8 (2020) 2156.
[49] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyper-parameter optimization, Adv. Neural Inf. Process. Syst. 24 (2011).
[50] J. Bergstra, D. Yamins, D. Cox, Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures, in: International Conference on Machine Learning, 2013, pp. 115–123.
[51] G. Boente, A. Pires, I. Rodrigues, Influence functions and outlier detection under the common principal components model: A robust approach, Biom. 89 (2002) 861–875.
[52] J. Bardet, S. Dimby, A new non-parametric detector of univariate outliers for distributions with unbounded support, Extremes 20 (2017) 751–775.
[53] N. Archana, S. Pawar, Periodicity detection of outlier sequences using constraint based pattern tree with mad, Int. J. Adv. Stud. Comput. Sci. Eng. 4 (2015) 34.
[54] J. Beirlant, Y. Goegebeur, J. Segers, J. Teugels, Statistics of Extremes: Theory and Applications, John Wiley & Sons, 2006.
[55] B. Courty, V. Schmidt, S. Luccioni, MarionCoutarel Goyal-Kamal, B. Feld, J. Lecourt, Saboni A. LiamConnell, Supatomic Inimaz, M. Léval, L. Blanche, A. Cruveiller, Zhao F. Ouminasara, A. Joshi, A. Bogroff, H. Lavoreille, N. Laskaris, E. Abati, D. Blank, Z. Wang, A. Catovic, M. Alencon, M. Stechly, C. Bauer, L. Araújo, JPW & MinervaBooks Mlco2/Codecarbon: V2.4.1, Zenodo, 2024, http://dx.doi.org/10.5281/zenodo.11171501.



Lorenzo Porcelli is a Ph.D. student in Computer Science at the University of Salerno, Italy, with research interests in anomaly detection, deep learning, and privacy. He holds B.Sc. (2019) and M.Sc. (2021) degrees in Computer Science from the same university. Since 2022, he has been a Technical Team member in the Operations directorate at the Bank of Italy's Directorate General for Information Technology, where he currently serves as Site Reliability Engineer and AIML Applied Research Scientist for the TIPS instant payment system managed by the Bank of Italy on behalf of the Eurosystem.

Marcello Trovati is a Professor in Computer Science in the Department of Computer Science, Edge Hill University, as well as the Editor in Chief of Applied Artificial Intelligence, Taylor&Francis. After having obtained his Ph.D. in Mathematics at the University of Exeter in 2007, specializing in theoretical dynamical systems with singularities, he has worked both in R&D and academic institutions. In 2016 Marcello joined the Computer Science Department at Edge Hill University where he is involved in several research themes and projects.

Francesco Palmieri is a professor at the University of Salerno, Italy. His major research interests concern high-performance networking protocols and architectures, routing algorithms and network security. Previously he has been an associate professor at the University of Salerno, an assistant professor at the Second University of Naples, and the Director of the networking division of the Federico II University in Naples, Italy. He has been closely involved with the development of the Internet in Italy as a member of the Technical-Scientific Advisory Committee and of the CSIRT of the Italian NREN GARR. He published more than 300 articles in leading technical journals, books and conferences and currently serves as the editor-in-chief of an international journal and is part of the editorial board or associate editor of several other well-reputed ones.