Context-Aware Deep Time-Series Decomposition for Anomaly Detection in Businesses

Youngeun Nam[1[0009−0008−8333−6488]], Patara Trirat[1[0000−0002−0889−813X]], Taeyoon Kim[1], Youngseop Lee[2], and Jae-Gil Lee[1(✉)[0000−0002−8711−7732]]

[1] School of Computing, KAIST
{youngeun.nam, patara.t, tykimseoul, jaegil}@kaist.ac.kr
[2] Samsung Electronics Co., Ltd.
yseop.lee@samsung.com

Abstract. Detecting anomalies in time series has become increasingly challenging as data collection technology develops, especially in real-world communication services, which require contextual information for precise prediction. To address this challenge, researchers usually use time-series decomposition to reveal underlying patterns, e.g., trends and seasonality. However, existing decomposition-based anomaly detectors do not explicitly consider such *contextual information*, limiting their ability to correctly detect contextual cases. This paper proposes *Time-CAD*, a new *context-aware deep* time-series decomposition framework to detect anomalies for a more practical scenario in real-world businesses. We verify the effectiveness of the novel design for integrating contextual information into deep time-series decomposition through extensive experiments on four real-world benchmarks, demonstrating improvements of up to 46% in time-series aware F_1 score on average.

Keywords: Time-Series Decomposition · Time-Series Anomaly Detection · Context-Aware Decomposition · Deep Learning.

1 Introduction

Time-series anomaly detection (TSAD) aims to identify data instances that diverge significantly from the normal range. From a traditional perspective [16], an anomaly is an observation that deviates from other observations, leading to suspicion that it was generated by a different mechanism. For instance, the sudden increase in website traffic, which may be thrice the usual traffic, can be attributed to various reasons such as competitor service breakdowns, natural disasters, or elections.

Accurately detecting anomalies is critical for corrective measures and potential damage prevention in real-world businesses. As an example, engineers use rich communication service traffic data to monitor the system status. Existing anomaly detectors commonly assume specific periodic patterns of time-series data; however, communication systems and businesses alike are often affected by unexpected events. Thus, we should *adaptively* detect anomalies based on ***contextual information***, such as days of the week and holidays, in such systems because what is normal on weekdays could be anomalous on weekends. Considering the context when detecting anomalies in time series is essential for precise prediction, which can be referred to as *contextual* anomaly detection [13].

For contextual anomaly detection, using time-series decomposition has several advantages over raw time series. First, directly extracting meaningful features from high-dimensional time series is challenging due to convoluted patterns. Second, accurate decomposition reveals the underlying trends, seasonalities, and noises, which helps better understand time series characteristics. Third, time-series decomposition eliminates the need for an overly complex neural network often required when using raw data; meanwhile, it improves the robustness of downstream tasks [35].

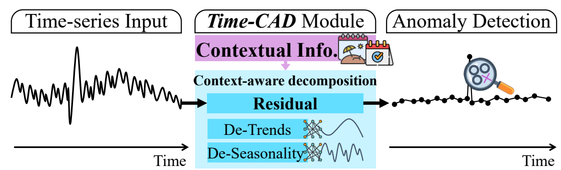


Fig. 1: Conceptual idea of *Time-CAD*.

Nevertheless, since the existing methods of time-series decomposition rely solely on statistical processes, the results are often overfitted to a particular time series [37, 38]. Furthermore, time series may not be decomposed appropriately as they do not analyze the temporal components individually based on irregular contextual information, such as spontaneous events and holidays, resulting in a high percentage of false alarms. Motivated by these limitations, as depicted in Fig. 1, we propose a novel ***Time-CAD*** framework designed specifically for anomaly detection to address the complexities and irregularities within the real-world time-series data by addressing the following challenges.

Challenge 1: *How to properly integrate contextual information into time-series decomposition?* Previous decomposition methods are based on statistical values such as the mean, median, or moving average. Consequently, they fail to adapt to abrupt trends or seasonal changes caused by events and holidays. Here, we inject sparse but informative variables—*contextual information*—to improve the robustness of decomposition results and use a simple neural network to increase the accuracy for a particular context.

Challenge 2: *How to extract normal patterns from the time series so that the residuals accurately represent potential anomalies?* Even Prophet [34], the only time-series decomposition method that considers auxiliary information, fails to extract accurate residuals due to the post-processing of auxiliary information after traditional decomposition. That is, the post-processing cannot explicitly infer irregular temporal information. Thus, we directly *inject* the contextual information into the decomposition process to accurately extract meaningful residuals for anomaly detection.

To overcome these two challenges, the main contributions of this paper are summarized as follows.

- We propose *Time-CAD*, the first *context-aware deep* time-series decomposition model designed for anomaly detection, that is robust to aperiodic patterns by explicitly considering contextual information.
- We show that *Time-CAD* produces flawless residuals and faithful normal patterns using only a simple neural network, thus, reducing false alarms.
- We demonstrate that *Time-CAD* improves TSAD performance on several real-world benchmarks through a series of experiments and verify its usability as a detector-agnostic framework by incorporating it with other anomaly detectors to enhance their detection accuracy.

2 Related Work

This section briefly discusses several TSAD methods based on the presence of time-series decomposition. For extensive reviews, see recent surveys [4, 9, 11].

2.1 Anomaly Detection *without* Decomposition

Without decomposition, we can classify TSAD into statistical and machine learning approaches. The most popular *statistical* approaches are regressive models [7,27], such as AutoRegressive Integrated Moving Average (ARIMA). They serve as a reference for effective statistical methods in time-series analysis. ARIMA [5] calculates the deviation of the predicted values from the observed values to solve the non-stationary problem in detecting anomalies after fitting the model. However, these models are sensitive to abrupt changes in time series.

Alternatively, Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [12] and One-Class Support Vector Machine (OCSVM) [28] are the popular *machine learning* methods. DBSCAN is a clustering-based anomaly detection method that classifies the data points into a core, border, or anomalous point. The anomalies are the data points that do not belong to any cluster. OCSVM is a non-linear one-class classification method that leverages the SVM trained on one particular class, i.e., normal instances. Anomalies will be determined if new samples do not belong to the class that is trained.

Recently, many studies [30, 40, 46, 48] have shown the superiority of *deep learning* for TSAD over traditional machine learning algorithms. Among several techniques [9], reconstruction-based [15,47] models are the most well-established approach and have consistently reported state-of-the-art performance. Donut [39], OmniAnomaly [33], and InterFusion [23] commonly adopted VAE-based models with additional mechanisms, such as Markov Chain Monte Carlo imputation, to improve detection accuracy. Similarly, USAD [2] and RANSynCoders [1] also adopted reconstruction-based models but with more simple architectures to enhance training and inference efficiency. To increase the detection performance with more learning capability on raw multivariate time series, more recent studies [36, 40] also proposed complex Transformer-based architectures.

Nevertheless, we argue that the overly complex neural architectures are unnecessary if we use time-series decomposition together in TSAD, as demonstrated by *Time-CAD* using simple reconstruction-based autoencoder models.

2.2 Anomaly Detection *with* Decomposition

Methods in this family mostly conduct time-series decomposition before anomaly detection. Twitter [17] developed a Seasonal Hybrid Extreme Studentized Deviate (S-H-ESD) algorithm. It uses robust statistics of median absolute deviation and generalized extreme Studentized deviate test after the decomposition process to detect anomalies. However, S-H-ESD has low anomaly detection quality in time series with high frequency, abrupt drop, and flat characteristics [37, 38]. At Facebook [34], time-series forecasting was performed through the model fitting with trends, seasonalities, holidays, and residuals as the results from time-series decomposition. Likewise, Microsoft [26] proposed an anomaly detection method that decomposes the time series and extracts spectral residuals using the Fourier transform-based algorithm. Lately, Alibaba [45] proposed a time-frequency analysis-based TSAD model by utilizing both time and frequency domains with decomposition and augmentation mechanisms to improve performance and interpretability.

Still, these studies heavily depend on statistical decomposition methods; thus, they can easily overfit a specific time-series dataset. Besides, the decomposition will not accurately work because temporal components that should be addressed differently—*sporadic contextual information*—are not considered, leading to a high rate of false alarms.

3 The *Time-CAD* Framework

This section presents the problem definition and details of ***Time***-series anomaly detection with ***C***ontext-***A***ware ***D***eep decomposition. Hence, ***Time-CAD***.

3.1 Problem Definition

Time-Series Decomposition Let $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N\} \in \mathbb{R}^{N \times M}$ be a time series of length N and $\mathcal{D}(\mathbf{x}_t)$ be a *decomposition* algorithm, where M is the number of variables[3] (or features). Thus, we denote the values at timestamp t as $\mathbf{x}_t = \{x_t^1, x_t^2, \ldots, x_t^M\}$. Since $\mathbf{x}_t$ can be expressed by a combination of trend-cycle τ_t, seasonality s_t, and residual r_t components, we then have $\mathcal{D}(\mathbf{x}_t) = \tau_t + s_t + r_t$ or $\mathcal{D}(\mathbf{x}_t) = \tau_t \times s_t \times r_t$ for additive or multiplicative decomposition, respectively.

Time-Series Anomaly Detection (TSAD) Let $W_t = \{\mathbf{x}_{t-w+1}, \ldots, \mathbf{x}_{t-1}, \mathbf{x}_t\}$ be a sliding window of length w at time t. Thus, we reformulate $\mathcal{X}$ as a sequence of overlapping windows $\mathcal{W} = \{W_1, W_2, \ldots, W_{N-w+1}\}$ used to train a TSAD model g_ϕ. The goal is to assign an anomaly label $y_t \in \{0, 1\}$ for each test data point $\hat{\mathbf{x}}_t \in \hat{W}_t$, where $\hat{W}_t \notin \mathcal{W}$, based on anomaly scores $\mathcal{A}_t$. If $\mathcal{A}_t$ exceeds a predefined threshold δ, $y_t = 1$; otherwise 0.

[3] A univariate time series is a special case of a multivariate time series when $M = 1$.

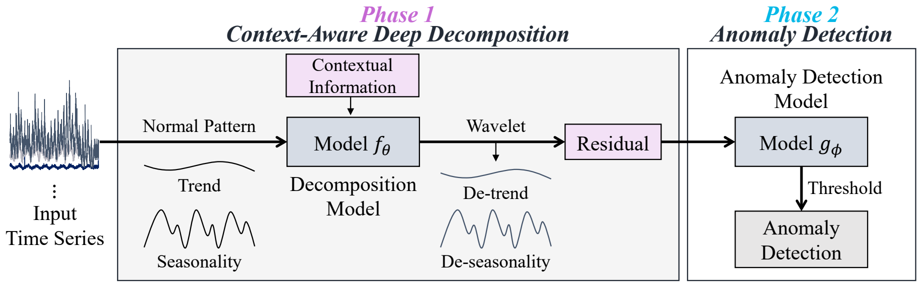


Fig. 2: Overview of the *Time-CAD* framework.

TSAD with Time-Series Decomposition Given a set of windows $\mathcal{W}$, we perform the time-series decomposition $\mathcal{D}(W_t)$ to obtain τ_t, s_t, and r_t for each window $W_t \in \mathcal{W}$. Then, we input the residual component r_t extracted from the window W_t to the TSAD model g_ϕ. Consequently, the model g_ϕ computes the anomaly scores $\mathcal{A}_t$ of all residuals r_t.

3.2 Overall Framework

Fig. 2 illustrates the overall *Time-CAD* framework consisting of two phases: *context-aware deep decomposition* and *anomaly detection*.

3.3 Phase 1: Context-Aware Deep Decomposition

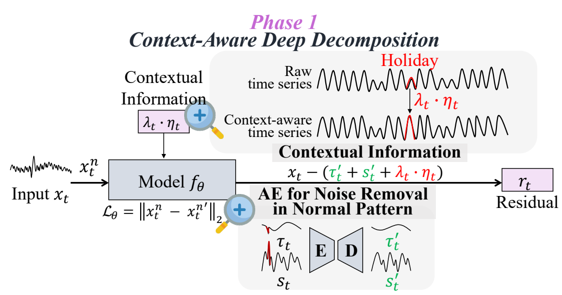


Fig. 3: Illustration of context-aware deep decomposition process (Phase 1).

As in Fig. 3, we train a neural network to extract normal patterns of time series, i.e., the trend and seasonality. Then, only the actual remainders or noises are left as the *residual*, which is the main focus of this process.

In this work, we employ the STL [10] algorithm as $\mathcal{D}(W_t)$ to initially decompose time series into $\tau_t + s_t + r_t$. Since we will use only *normal patterns* x_t^n to train a decomposition model f_θ, $x_t^n = \tau_t + s_t$. In particular, we use a Gated Recurrent Unit (GRU) [8]-based autoencoder as the decomposition model f_θ. The output of f_θ is the reconstructed normal pattern $x_t^{n'}$. Thus, f_θ minimizes the reconstruction errors between the decomposed time-series normal patterns x_t^n and their reconstructed versions $x_t^{n'}$ with loss $\mathcal{L}_\theta = \|x_t^n - x_t^{n'}\|_2$.

Algorithm 1 Training Algorithm of *Time-CAD*

INPUT: Normal windows dataset $\mathcal{W} = \{W_1, \cdots, W_T\}$, contextual information η, hyperparameter λ_t
OUTPUT: Trained f_θ, g_ϕ

$\theta, \phi \leftarrow$ initialize weights;
/* train deep decomposition model */
for $epoch = 1$ **to** $epochs^{\text{decomposition}}$ **do**
  for $t = 1$ **to** $t = T$ **do**
    $\tau_t \leftarrow$ trend component of $\mathcal{D}(W_t)$;
    $s_t \leftarrow$ seasonal component of $\mathcal{D}(W_t)$;
    /* reconstruct normal pattern */
    $W_t^{n'} \leftarrow f_\theta(\tau_t + s_t)$;
    $\mathcal{L}_\theta \leftarrow \|W_t^n - W_t^{n'}\|_2$;
    $\theta \leftarrow$ update weights using $\mathcal{L}_\theta$;
    $r_t \leftarrow \Psi(W_t - f_\theta(\tau_t + s_t) + \lambda_t \cdot \eta_t)$;
  end for
end for
/* train anomaly detection model */
for $epoch = 1$ **to** $epochs^{\text{detection}}$ **do**
  for $t = 1$ **to** $t = T$ **do**
    $r_t{}' \leftarrow g_\phi(r_t)$;
    $\mathcal{L}_\phi \leftarrow \|r_t - r_t{}'\|_2$;
    $\phi \leftarrow$ update weights using $\mathcal{L}_\phi$;
  end for
end for
return f_θ, g_ϕ

After training the decomposition model f_θ, we remove the normal pattern $x_t^{n'}$ from the original time series x_t. Here, we use temporal contextual information to regulate the residuals so that the algorithm recognizes the contextual information. Note that other contextual information (e.g., sensor location) can also be used in this framework depending on the specific application domains. The temporal contextual information includes whether the timestamp is a *weekend*, *holiday*, *day before holiday*, and specific *event*. Formally, the contextual information vector $\eta_t = [Z(t); \mathbf{1}(t \in \mathcal{H})]$, where $Z(t)$ is the seasonal information including the hour, month, and year, $\mathcal{H}$ is the list of holidays. For each time t, we can additionally control λ_t depending on the requirements of each application domain. $\lambda_t \cdot \eta_t = \lambda_t \cdot [Z(t); \mathbf{1}(t \in \mathcal{H})]\ \forall t \in \{1, \ldots, N\}$, where λ_t is the hyperparameter denoting the degree of contextual information. Finally, we apply Wavelet transform to remove trifling signal noises. As a result, the **final remaining residual** is formulated by

$$r_t = \Psi(x_t - f_\theta(\tau_t + s_t) + \lambda_t \cdot \eta_t). \tag{1}$$

Notably, thanks to non-linear mapping, we find that our context-aware deep time-series decomposition is more robust and reliable than existing decomposition without deep learning. That is, $\Psi(x_t - \underline{f_\theta(\tau_t + s_t)} + \lambda_t \cdot \eta_t)$ is better than

Algorithm 2 Inference Steps of *Time-CAD*

INPUT: Test windows dataset $\widehat{\mathcal{W}} = \{\widehat{W}_1, \ldots, \widehat{W}_{\hat{T}}\}$, contextual information $\hat{\eta}$, hyperparameter λ_t, threshold δ
OUTPUT: Labels $y : \{y_1, \ldots, y_{\hat{T}}\}$
1: /* deep decomposition model */
2: **for** $t = 1$ **to** $\hat{T}$ **do**
3: $\hat{\tau_t} \leftarrow$ trend component of $\mathcal{D}(\widehat{W}_t)$;
4: $\hat{s_t} \leftarrow$ seasonal component of $\mathcal{D}(\widehat{W}_t)$;
5: $\hat{r_t} \leftarrow \Psi(W_t - f^*_\theta(\hat{\tau_t} + \hat{s_t}) + \lambda_t \cdot \hat{\eta}_t)$;
6: **end for**
7: /* anomaly detection model */
8: **for** $t = 1$ **to** $\hat{T}$ **do**
9: $\hat{r_t}' \leftarrow g_{\phi^*}(\hat{r_t})$;
10: $\mathcal{A}_t \leftarrow \|\hat{r_t} - \hat{r_t}'\|_2$;
11: **if** $\mathcal{A}_t > \delta$ **then**
12: $y_t \leftarrow 1$ /* identify as an anomalous value */
13: **else**
14: $y_t \leftarrow 0$ /* identify as a normal value */
15: **end if**
16: **end for**
17: **return** $y : \{y_1, \cdots, y_{\hat{T}}\}$

$\Psi(x_t - \underline{(\tau_t + s_t)} + \lambda_t \cdot \eta_t)$ because the trend τ_t and seasonality s_t initially extracted by the STL decomposition has the following limitation. While the STL decomposition cannot ideally extract the trend and seasonality when the raw time series has noises and potential contamination of anomalies in the training data, our model f_θ eliminates the noise and potential contamination by the denoising autoencoder, resulting in a more robust normal pattern. As in Fig. 6, we empirically verify the effectiveness of the proposed *deep* decomposition model.

Lines 3–13 of Algorithm 1 and Lines 2–6 of Algorithm 2 summarize the process of this phase.

3.4 Phase 2: Time-Series Anomaly Detection

In this phase, we use the derived residuals, Eq. (1) in Phase 1 (§3.3), as the input features for an anomaly detection model. Here, we train the TSAD model g_ϕ to reconstruct the residuals of *normal* cases. If the residuals of anomalous instances are input to the detection model, the model will give high reconstruction errors. We later use these reconstruction errors as anomaly scores $\mathcal{A}_t$. Fig. 4 visualizes the process of this phase.

In this work, we use a bidirectional GRU autoencoder network [29] as the anomaly detection model g_ϕ. The input is fed with the overlapping sliding window $W_t \in \mathcal{W}$, where $W_t = \{r_{t-w+1}, \ldots, r_{t-1}, r_t\}$. Accordingly, we train the anomaly detection model g_ϕ by minimizing the reconstruction loss $\mathcal{L}_\phi =$

$\|r_t - r_t{}'\|_2$ between the original r_t and reconstructed residuals $r_t{}'$. During the inference, the anomaly score $\mathcal{A}_t$ is computed by the reconstruction errors. Hence, $\mathcal{A}_t = \|\hat{r_t} - \hat{r_t}{}'\|_2$, where $\hat{r_t}$ is an unseen residual of a new time series and $\hat{r_t}{}'$ is a reconstructed residual. If the anomaly score $\mathcal{A}_t$ at time t is greater than a predefined threshold δ, it is determined as an anomaly (i.e., 1); otherwise normal (i.e., 0). Although we use simple bidirectional GRU autoencoders in *Time-CAD*, any other architectures or models can also be used.

Ideally, when a time-series value x_t is significantly diverse from the learned normal patterns, the detection model should correctly identify it as an anomaly. To achieve this, unlike previous studies, we thus use the residual r_t as the input to the anomaly detection model instead of the raw time series. The underlying reason is that the residual—a remainder of the de-trend and de-seasonality process—is associated with abnormality or noise. Therefore, the model can detect the anomalies with much simpler input, yet achieve higher accuracy. Lines 15–21 of Algorithm 1 and Lines 8–16 of Algorithm 2 outline the process of this phase.

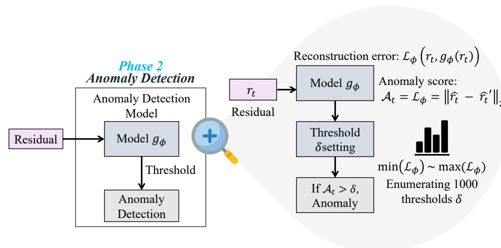


Fig. 4: Illustration of time-series anomaly detection process (Phase 2).

4 Evaluation

In this section, we design the experiments to answer the following questions:

Q1 How well *Time-CAD* performs TSAD compared with baseline methods?
Q2 How effective is the *context-aware deep* decomposition?
Q3 Is *Time-CAD* feasible to be deployed in production?

The source code is available at https://github.com/kaist-dmlab/Time-CAD.

4.1 Experimental Setup

Data Description As summarized in Table 1, we use four real-world benchmarks containing *seven* dataset entities to comprehensively evaluate the anomaly detection performance on diverse businesses and industries. **KPI**[4] is a single-entity key performance indicator dataset used in a competition. It measures the quality of Internet services. **Energy**[5] benchmark measures the health status of power equipment. This benchmark has two datasets of the different lengths and is

[4] https://github.com/NetManAIOps/KPI-Anomaly-Detection
[5] https://aihub.or.kr/aihubdata/data/list.do

Table 1: Benchmark statistics.

Datasets	**Collection Date** (DD.MM.YYYY)	# **Timestamp**	# **Train**	# **Test**	**Entity×Dim.**	# **Anomaly**
RCS	01.02.2021 – 01.04.2022	34,902	21,600	13,302	3 × 8	160 (0.46%)
KPI	31.07.2017 – 30.10.2017	111,370	66,822	44,548	1 × 1	1,102 (0.99%)
Energy	13.11.2020 – 16.12.2020	47,003	41,654	5,349	2 × 32	2,772 (5.90%)
IoT-Modbus	01.04.2019 – 25.04.2019	51,106	15,332	35,774	1 × 4	16,106 (31.51%)

collected from 450 facilities on a minute-interval basis for 30 days. Each instance in the test set is labeled with normal, caution, or warning status. **IoT-Modbus**[6] is a public single-entity benchmark from an Internet of Things system. The data is collected from realistic and large-scale networks having four features indicating Modbus function code: an input register, a discrete value, a holding register, and a coil. Anomalous labels are DoS, DDoS, and backdoor attacks [25]. **RCS** is a private benchmark complied by a cloud operation group at a mobile business company measuring rich communication service traffic records, e.g., the number of sent and received text messages. This benchmark consists of three datasets of the same lengths and is collected every ten minutes.

Evaluation Metric and Threshold Setting We adopt time-series aware precision-recall metrics [19], TaPR, specifically designed for TSAD tasks to reflect the feature of a *series of instances*. Since the conventional point-wise metrics overlook the characteristics of a series of instances, they suffer from a scarcity of evaluating the variety of the detected anomalies. At the same time, the widely-used point-adjust metric suffers from overestimation issues [1,15]. Therefore, we assess the performance with TaPR and the corresponding F_1 scores: TaF_1.

To identify anomalies during testing, we enumerate $1,000$ thresholds δ distributed uniformly from the minimum to the maximum of the anomaly scores $\mathcal{A}_t$ for all timestamps t in the test data to avoid highly relying on the threshold policy [31,41]. Moreover, in practice, it is more important to have an excellent F_1 metric at a certain threshold than a generally good result [14]. Thus, we report the *best* TaF_1 based on the optimal threshold of each model.

Comparison Baselines We compare *Time-CAD* to both traditional and recent state-of-the-art methods with and without time-series decomposition as follows.

Traditional Methods.

1. Local Outlier Factor (**LOF**) [6] is an unsupervised outlier detector that measures the local deviation of the density of a given sample to its neighbors.
2. Isolation Forest (**ISF**) [24] is a well-known anomaly detection algorithm that works on the principle of isolating anomalies using tree-based structures.

[6] https://research.unsw.edu.au/projects/toniot-datasets

Table 2: Performance comparison between anomaly detection models in the best TaF_1 with the highest scores highlighted in **bold**.

Datasets	RCS-1	RCS-2	RCS-3	KPI	Energy-1	Energy-2	IoT-Modbus	Avg. ↑	Rank ↓
Non-Decomposition									
LOF	0.474 (±0.00)	0.422 (±0.00)	0.434 (±0.00)	0.177 (±0.00)	0.701 (±0.00)	0.973 (±0.00)	0.701 (±0.00)	0.555	12
ISF	0.614 (±0.00)	0.745 (±0.00)	0.458 (±0.00)	0.823 (±0.00)	0.809 (±0.00)	0.975 (±0.00)	0.642 (±0.00)	0.724	9
OCSVM	0.619 (±0.00)	0.292 (±0.00)	0.562 (±0.00)	0.531 (±0.00)	0.954 (±0.00)	0.946 (±0.00)	0.690 (±0.00)	0.656	10
AE	0.472 (±0.08)	0.583 (±0.10)	0.435 (±0.02)	0.861 (±0.00)	0.954 (±0.00)	0.976 (±0.00)	0.894 (±0.00)	0.739	8
MS-RNN	0.514 (±0.02)	0.740 (±0.01)	0.484 (±0.01)	0.915 (±0.01)	0.954 (±0.00)	0.979 (±0.01)	0.826 (±0.05)	0.773	6
OmniAnomaly	0.503 (±0.00)	0.710 (±0.01)	0.922 (±0.00)	0.892 (±0.01)	0.950 (±0.00)	0.980 (±0.00)	0.762 (±0.01)	0.774	4
RANSynCoders	0.435 (±0.01)	0.613 (±0.01)	0.425 (±0.01)	0.227 (±0.03)	0.914 (±0.01)	0.986 (±0.01)	**0.987** (±0.01)	0.655	11
TranAD	0.461 (±0.02)	0.941 (±0.00)	0.544 (±0.11)	0.934 (±0.04)	0.953 (±0.00)	0.915 (±0.06)	0.664 (±0.01)	0.773	5
Decomposition									
AE-STL	0.867 (±0.02)	0.885 (±0.02)	0.911 (±0.03)	0.922 (±0.02)	0.936 (±0.02)	**0.987** (±0.01)	0.894 (±0.00)	0.915	2
SR-CNN	0.547 (±0.00)	0.733 (±0.00)	0.594 (±0.00)	0.488 (±0.00)	0.952 (±0.00)	0.959 (±0.00)	0.977 (±0.00)	0.750	7
TFAD	0.539 (±0.02)	0.632 (±0.00)	0.762 (±0.11)	0.854 (±0.04)	0.956 (±0.00)	0.955 (±0.07)	0.886 (±0.01)	0.798	3
Time-CAD	**0.944** (±0.00)	**0.955** (±0.00)	**0.944** (±0.00)	**0.937** (±0.00)	**0.961** (±0.01)	0.986 (±0.00)	0.957 (±0.00)	**0.955**	**1**

3. **OCSVM** [28] is an unsupervised outlier detection algorithm based on SVM. It maximizes the margin between the origin and the normality and defines the decision boundary as the hyper-plane that determines the margin.
4. Autoencoder (**AE**) [3] is a simple neural architecture that uses the symmetrical encoder and decoder network for anomaly detection. Anomaly scores are the differences between the inputs and reconstructed outputs.
5. Autoencoder with STL decomposition (**AE-STL**) [10] is a combination of the AE and the traditional time-series decomposition method, STL. The residuals from STL are input to AE instead of the raw time series.

State-of-the-art Models.

6. Modified-RNN (**MS-RNN**) [21] is a modified version of an anomaly detector that exploits sparsely-connected recurrent neural networks (RNNs) and an ensemble of sequence-to-sequence AE for multi-resolution learning.
7. **SR-CNN** [26] is a time-series decomposition-based anomaly detector. It uses spectral residual to extract saliency maps and use them as input for convolutional neural networks to detect anomalies.
8. **OmniAnomaly** [33] is a GRU-based VAE that captures complex temporal dependency between multivariate time series and maps observations to stochastic variables.
9. **RANSynCoders** [1] utilizes pre-trained AE to extract primary frequencies across the signals on the latent representation for synchronizing time series.
10. **TranAD** [36] is a Transformer-based model that uses attention-based sequence encoders to perform inference with broader temporal trends in time series. It uses focus score-based self-conditioning to enable robust multimodal feature extraction and adversarial training to gain stability.
11. **TFAD** [45] is a time-frequency analysis-based anomaly detection model that utilizes both time and frequency domains to improve performance in anomaly detection. The model incorporates time series decomposition and data augmentation mechanisms to enhance performance and interpretability.

Table 3: Performance comparison between the different decomposition methods in the best TaF_1 with the highest scores highlighted in **bold**.

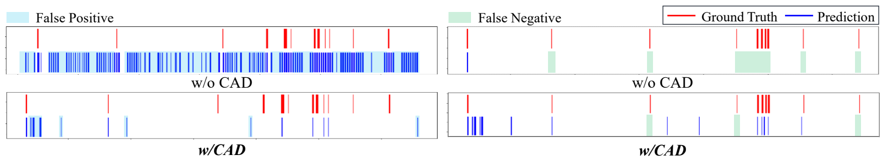
Datasets	***Time-CAD***			MS-RNN		OmniAnomaly	
	w/CAD	w/o CAD	w/o DNN	**w/CAD**	w/o CAD	**w/CAD**	w/o CAD
RCS-1	**0.944** ($\pm$0.00)	0.633 ($\pm$0.00)	0.871 ($\pm$0.00)	**0.789** ($\pm$0.00)	0.514 ($\pm$0.09)	**0.939** ($\pm$0.00)	0.503 ($\pm$0.00)
RCS-2	**0.955** ($\pm$0.00)	0.710 ($\pm$0.01)	0.886 ($\pm$0.00)	**0.829** ($\pm$0.00)	0.740 ($\pm$0.01)	**0.949** ($\pm$0.01)	0.710 ($\pm$0.01)
RCS-3	**0.944** ($\pm$0.00)	0.622 ($\pm$0.00)	0.858 ($\pm$0.01)	**0.785** ($\pm$0.01)	0.484 ($\pm$0.01)	**0.938** ($\pm$0.01)	0.662 ($\pm$0.00)
KPI	**0.937** ($\pm$0.00)	0.905 ($\pm$0.00)	0.936 ($\pm$0.01)	**0.916** ($\pm$0.01)	0.915 ($\pm$0.01)	**0.915** ($\pm$0.01)	0.892 ($\pm$0.01)
Energy-1	**0.961** ($\pm$0.01)	0.953 ($\pm$0.00)	0.953 ($\pm$0.01)	0.927 ($\pm$0.00)	**0.954** ($\pm$0.00)	**0.953** ($\pm$0.00)	0.950 ($\pm$0.00)
Energy-2	0.986 ($\pm$0.00)	**0.989** ($\pm$0.00)	0.986 ($\pm$0.00)	**0.980** ($\pm$0.00)	0.979 ($\pm$0.01)	**0.986** ($\pm$0.00)	0.980 ($\pm$0.00)
IoT-Modbus	**0.957** ($\pm$0.00)	0.762 ($\pm$0.00)	0.894 ($\pm$0.01)	**0.841** ($\pm$0.00)	0.826 ($\pm$0.05)	**0.942** ($\pm$0.00)	0.762 ($\pm$0.01)

4.2 Performance Comparison

Anomaly Detection Performance (Q1) Table 2 presents the overall performance in the best TaF_1 metric. We run each model three times to ensure reproducibility and avoid occasional results, then report the average and standard deviation. *Time-CAD* demonstrates state-of-the-art performance in most datasets except for Energy and IoT-Modbus. On average, *Time-CAD* outperforms all baselines by a significant margin (up to 46%), especially on the RCS datasets expected to be strongly affected by the temporal contextual conditions. On the other hand, as Energy and IoT-Modbus datasets are machinery data that do not directly associate with people, they show a regular pattern regardless of the temporal contexts. Thus, we conjecture that other types of contextual information, such as spatial or environmental information, will further enhance the detection performance on Energy and IoT-Modbus datasets.

Ablation and Case Study (Q2) To examine the contributions of the *context-aware deep* decomposition (CAD), we perform ablation studies on both the proposed *Time-CAD* and the baselines. As presented in Table 3, w/CAD denotes the presence of context-aware deep decomposition, while w/o CAD is the absence. Likewise, w/o DNN indicates the context-aware decomposition but *without* the deep neural network (DNN) model f_θ designed to evaluate the effect of DNN in the decomposition process. According to the results, it is evident that w/CAD performs significantly better than w/o CAD and w/o DNN counterparts in most datasets. Therefore, we ascertain that *Time-CAD* can significantly boost the TSAD performance of any anomaly detectors, demonstrating its high usability as a model-agnostic framework.

Additionally, Fig. 5 depicts anomaly detection results where red lines indicate the ground truths and the blue lines are prediction results on RCS-1 (Fig. 5a) and RCS-3 (Fig. 5b) datasets. For each dataset, the upper plot shows anomaly detection without *Time-CAD*, and the lower plot shows anomaly detection with *Time-CAD*. In **RCS-1**, the upper plot illustrates many false positives while the lower one adequately detects anomalies. In contrast, for **RCS-3**, the upper plot



(a) RCS-1 (b) RCS-3

Fig. 5: Visualization of labels (red) and predicted anomalies (blue).

has many false negatives, while the lower sufficiently detects anomalies, albeit with a few errors.

Lastly, we visually compare the extracted *residual components* between the different decomposition methods. As in Fig. 6, we consider a festival from September $20th$ to $22nd$ as a case study. The first plot exhibits different patterns of the **RCS** time series, yet within normal ranges. Unfortunately, without the contextual information and deep neural network, the second plot shows that the original STL decomposition cannot decompose the valid residual components, causing an increase in false positives when detecting anomalies. In the third plot, the residual components during the festival time are relieved thanks to the *contextual information*. Still, it has noises that may adversely affect detection performance. Finally, the fourth plot demonstrates the advantage of *Time-CAD* in precisely extracting normal patterns during the distinct period, resulting in meaningful residuals for anomaly detection. Compared to the without DNN counterpart, the results confirm that the *deep* decomposition yields more ideal residuals by robustly detaching normal patterns, thus, mitigating false positives.

4.3 Deployment Feasibility

As an answer to **Q3**, we study the feasibility of *Time-CAD* in detection quality and computation time aspects on the real-world **RCS** datasets that contain several business metrics.

Detection Quality After the offline training on about 5-month multivariate time-series datasets, *Time-CAD* detects nearly all anomalies in 3-month testing data with a strong performance of 0.948 in TaF_1 on average across three datasets.

Computation Time We run the inference phase on a server equipped with an NVIDIA GeForce GTX 3090Ti. *Time-CAD* takes only about 69 seconds for each 3-month-long dataset that contains about 13K instances, meaning that it takes only *5 milliseconds* for a single timestamp. Hence, *Time-CAD* is feasible to detect anomalies in a real-time environment.

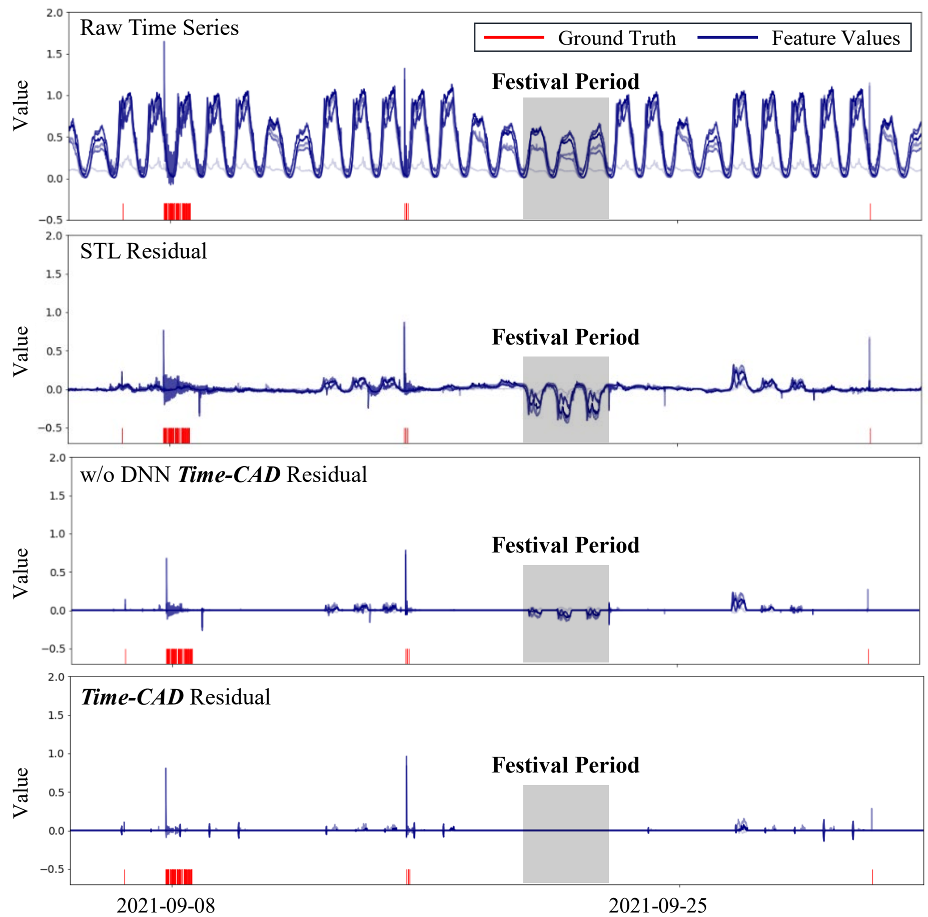


Fig. 6: Comparison of the residual components between the different decomposition methods on **RCS** dataset.

System Prototype As a production prototype, we make a pilot deployment with the trained *Time-CAD* detection model to detect anomalies in an online batch-based web application[7] by connecting it with a real-time database. Once the time-series instances are satisfied with a predefined window size, the system will run the detection model and return the anomaly scores for all timestamps along with the original time series to facilitate users for a quick inspection and interpretation in which locations potential anomalies have occurred.

[7] https://time-cad.web.app

5 Conclusion

This paper introduces a novel *context-aware deep* time-series decomposition framework for anomaly detection called *Time-CAD*. With the collaboration of deep learning and contextual information, we show that *Time-CAD* accurately extracts a clear periodic pattern by enhancing the properties of each component in a time series, leading to an improvement in anomaly detection performance by up to 46%. Empirically, the proposed framework demonstrates its superiority over state-of-the-art methods on four benchmarks in the time-series aware F_1 metric. We further verify that context-aware deep decomposition explicitly adapts to aperiodic patterns by using contextual information through a series of ablation studies. Finally, we expect the proposed *Time-CAD* framework to advance the development of anomaly detectors with different types of contextual information, which is crucial for various application domains and businesses.

Ethical Statement This work adheres to ethical standards and guidelines for scientific research. We use publicly available datasets and obtain all necessary permissions and approvals before conducting the experiments and data collection. Therefore, we ensure the privacy and anonymity of all human participants involved in the data collection process. In particular, the RCS and KPI datasets are the communication service datasets significantly associated with real users. Both RCS and KPI datasets were completely anonymized with their types and features before we received them. Our research aims to advance the field of anomaly detection having critical applications in various domains, such as finance, healthcare, and cyber security. However, there might be potential malicious impacts when inappropriately using our work. For example, the advancement and findings from *Time-CAD* might be adversely exploited for devising more subtle and sophisticated attacks or deceptions.

Acknowledgements This work was partly supported by Mobile eXperience Business, Samsung Electronics Co., Ltd. (Real-time Service Incident Prediction Development) and the National Research Foundation of Korea (NRF) grant funded by the Korea government (Ministry of Science and ICT) (No. 2023R1A2C 2003690).

References

1. Abdulaal, A., Liu, Z., Lancewicki, T.: Practical approach to asynchronous multivariate time series anomaly detection and localization. In: ACM SIGKDD (2021)
2. Audibert, J., Michiardi, P., Guyard, F., Marti, S., Zuluaga, M.A.: USAD: Unsupervised anomaly detection on multivariate time series. In: ACM SIGKDD (2020)
3. Bergmann, P., Löwe, S., Fauser, M., Sattlegger, D., Steger, C.: Improving unsupervised defect segmentation by applying structural similarity to autoencoders. arXiv:1807.02011 (2018)

4. Blázquez-García, A., Conde, A., Mori, U., Lozano, J.A.: A review on outlier/anomaly detection in time series data. ACM CSUR **54**(3) (2021)
5. Braei, M., Wagner, S.: Anomaly detection in univariate time-series: A survey on the state-of-the-art. arXiv:2004.00433 (2020)
6. Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: Identifying density-based local outliers. In: ACM SIGMOD (2000)
7. Chandola, V.: Anomaly Detection for Symbolic Sequences and Time Series Data. University of Minnesota (2009)
8. Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y.: Learning phrase representations using rnn encoder-decoder for statistical machine translation. arXiv:1406.1078 (2014)
9. Choi, K., Yi, J., Park, C., Yoon, S.: Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. IEEE Access (2021)
10. Cleveland, R.B., Cleveland, W.S., McRae, J.E., Terpenning, I.: STL: A seasonal-trend decomposition. J. Off. Stat **6**(1) (1990)
11. Darban, Z.Z., Webb, G.I., Pan, S., Aggarwal, C.C., Salehi, M.: Deep learning for time series anomaly detection: A survey. arXiv:2211.05244 (2022)
12. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: ACM SIGKDD (1996)
13. Farshchi, M., Weber, I., Della Corte, R., Pecchia, A., Cinque, M., Schneider, J.G., Grundy, J.: Contextual anomaly detection for a critical industrial system based on logs and metrics. In: EDCC (2018)
14. Feng, C., Tian, P.: Time series anomaly detection for cyber-physical systems via neural system identification and bayesian filtering. In: ACM SIGKDD (2021)
15. Garg, A., Zhang, W., Samaran, J., Savitha, R., Foo, C.S.: An evaluation of anomaly detection and diagnosis in multivariate time series. IEEE Trans. Neural Netw. Learn. Syst. (2021)
16. Hawkins, D.M.: Identification of Outliers, vol. 11. Springer (1980)
17. Hochenbaum, J., Vallis, O.S., Kejariwal, A.: Automatic anomaly detection in the cloud via statistical learning. arXiv:1704.07706 (2017)
18. Hodrick, R.J., Prescott, E.C.: Postwar us business cycles: An empirical investigation. J. Money Credit Bank. (1997)
19. Hwang, W.S., Yun, J.H., Kim, J., Kim, H.C.: Time-series aware precision and recall for anomaly detection: Considering variety of detection result and addressing ambiguous labeling. In: CIKM (2019)
20. Hyndman, R.J., Athanasopoulos, G.: Forecasting: Principles and Practice. OTexts (2018)
21. Kieu, T., Yang, B., Guo, C., Jensen, C.S.: Outlier detection for time series with recurrent autoencoder ensembles. In: IJCAI (2019)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv:1412.6980 (2014)
23. Li, Z., Zhao, Y., Han, J., Su, Y., Jiao, R., Wen, X., Pei, D.: Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In: ACM SIGKDD (2021)
24. Liu, F.T., Ting, K.M., Zhou, Z.H.: Isolation forest. In: IEEE ICDM (2008)
25. Moustafa, N.: A new distributed architecture for evaluating ai-based security systems at the edge: Network ton_iot datasets. SCS **72** (2021)
26. Ren, H., Xu, B., Wang, Y., Yi, C., Huang, C., Kou, X., Xing, T., Yang, M., Tong, J., Zhang, Q.: Time-series anomaly detection service at microsoft. In: ACM SIGKDD (2019)

27. Rousseeuw, P.J., Leroy, A.M.: Robust Regression and Outlier Detection, vol. 589. John wiley & sons (2005)
28. Schölkopf, B., Williamson, R.C., Smola, A.J., Shawe-Taylor, J., Platt, J.C., et al.: Support vector method for novelty detection. In: NeurIPS (1999)
29. Schuster, M., Paliwal, K.K.: Bidirectional recurrent neural networks. IEEE Trans. Signal Process. **45**(11) (1997)
30. Shen, L., Li, Z., Kwok, J.: Timeseries anomaly detection using temporal hierarchical one-class network. In: NeurIPS (2020)
31. Shen, L., Yu, Z., Ma, Q., Kwok, J.T.: Time series anomaly detection with multiresolution ensemble decoding. In: AAAI (2021)
32. Song, X., Wu, M., Jermaine, C., Ranka, S.: Conditional anomaly detection. IEEE TKDE **19**(5) (2007)
33. Su, Y., Zhao, Y., Niu, C., Liu, R., Sun, W., Pei, D.: Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In: ACM SIGKDD (2019)
34. Taylor, S.J., Letham, B.: Forecasting at scale. Am. Stat. **72**(1) (2018)
35. Theodosiou, M.: Forecasting monthly and quarterly time series using stl decomposition. Int. J. Forecast. **27**(4) (2011)
36. Tuli, S., Casale, G., Jennings, N.R.: TranAD: deep transformer networks for anomaly detection in multivariate time series data. VLDB Endow. **15**(6), 1201–1214 (2022)
37. Wen, Q., Gao, J., Song, X., Sun, L., Xu, H., Zhu, S.: RobustSTL: A robust seasonal-trend decomposition algorithm for long time series. In: AAAI (2019)
38. Wen, Q., Zhang, Z., Li, Y., Sun, L.: Fast RobustSTL: Efficient and robust seasonal-trend decomposition for time series with complex patterns. In: ACM SIGKDD (2020)
39. Xu, H., Chen, W., Zhao, N., Li, Z., Bu, J., Li, Z., Liu, Y., Zhao, Y., Pei, D., Feng, Y., et al.: Unsupervised anomaly detection via variational auto-encoder for seasonal KPIs in web applications. In: WWW (2018)
40. Xu, J., Wu, H., Wang, J., Long, M.: Anomaly transformer: Time series anomaly detection with association discrepancy. In: ICLR (2022)
41. Yoo, Y.H., Kim, U.H., Kim, J.H.: Recurrent reconstructive network for sequential anomaly detection. IEEE CYB (2019)
42. Yoon, S., Lee, J., Lee, B.S.: NETS: extremely fast outlier detection from a data stream via set-based processing. VLDB Endow. **12**(11) (2019)
43. Yoon, S., Lee, J., Lee, B.S.: Ultrafast local outlier detection from a data stream with stationary region skipping. In: ACM SIGKDD (2020)
44. Yoon, S., Shin, Y., Lee, J., Lee, B.S.: Multiple dynamic outlier-detection from a data stream by exploiting duality of data and queries. In: ACM SIGMOD (2021)
45. Zhang, C., Zhou, T., Wen, Q., Sun, L.: TFAD: A decomposition time series anomaly detection architecture with time-frequency analysis. In: CIKM. pp. 2497–2507 (2022)
46. Zhang, C., Song, D., Chen, Y., Feng, X., Lumezanu, C., Cheng, W., Ni, J., Zong, B., Chen, H., Chawla, N.V.: A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In: AAAI (2019)
47. Zhao, H., Wang, Y., Duan, J., Huang, C., Cao, D., Tong, Y., Xu, B., Bai, J., Tong, J., Zhang, Q.: Multivariate time-series anomaly detection via graph attention network. In: IEEE ICDM (2020)
48. Zhou, B., Liu, S., Hooi, B., Cheng, X., Ye, J.: BeatGAN: Anomalous rhythm detection using adversarially generated time series. In: IJCAI. pp. 4433–4439 (2019)