Contaminated Multivariate Time-Series Anomaly Detection with Spatio-Temporal Graph Conditional Diffusion Models

Thi Kieu Khanh Ho[1,2] **Narges Armanfard**[1,2]

[1]Department of Electrical and Computer Engineering, McGill University
[2]Mila - Quebec AI Institute, Montreal, QC, Canada

Abstract

Mainstream unsupervised anomaly detection algorithms often excel in academic datasets, yet their real-world performance is restricted due to the controlled experimental conditions involving clean training data. Addressing the challenge of training with noise, a prevalent issue in practical anomaly detection, is frequently overlooked. In a pioneering endeavor, this study delves into the realm of label-level noise within sensory time-series anomaly detection (TSAD). This paper presents a novel and practical TSAD when the training data is contaminated with anomalies. The introduced approach, called TSAD-C, is devoid of access to abnormality labels during the training phase. TSAD-C encompasses three modules: a Decontaminator to rectify anomalies present during training and swiftly prepare the decontaminated data for subsequent modules; a Long-range Variable Dependency Modeling module to capture long-range intra-variable and inter-variable dependencies within the decontaminated data that is considered as a surrogate of the pure normal data; and an Anomaly Scoring module that leverages insights of the first two modules to detect all types of anomalies. Our extensive experiments conducted on four reliable, diverse, and challenging datasets conclusively demonstrate that TSAD-C surpasses existing methods, thus establishing a new state-of-the-art in the TSAD field.

1 INTRODUCTION

Multivariate time-series data (MTS) is referred to as time-stamped data that consists of multiple variables, i.e., for each timestamp, there are multiple values associated with it. Time-series anomaly detection (TSAD) is the process of detecting unusual patterns or events within time-series data that deviate from the expected behavior Schmidl et al. [2022]. The unusual patterns can be found in many real-world applications such as instances of financial fraud, abrupt temperature spikes or unforeseen precipitation in weather data, security breaches, system malfunctions, and irregularities in brain activities. Many algorithms have been proposed to detect anomalies in time-series data Su et al. [2019], Audibert et al. [2020], Deng and Hooi [2021], Chen et al. [2022], Yang et al. [2023], Ho and Armanfard [2023], Chen et al. [2024], Sun et al. [2024], Fang et al. [2024]. However, several critical challenges remain unresolved.

First, existing studies can be categorized into two groups, which are supervised methods utilizing both labeled normal and abnormal data in the training phase Lai et al. [2024], and unsupervised methods assuming that only normal data is available during training Yang et al. [2023]. Given the demanding nature of accurately labeling anomalous patterns – proving to be time-consuming, costly, and labor-intensive – supervised approaches are deemed impractical. Consequently, there has been a recent surge in the development of unsupervised approaches. However, in many real-world applications, anomalies often sneak into normal data, which come from the data shift or human misjudgment Jiang et al. [2022]. These unsupervised methods are sensitive to the seen anomalies due to their exhaustive strategy to model the normal training data, hence, they would misdetect similar anomaly samples in the test phase. Therefore, developing a method that can detect anomalies while being trained on contaminated data is necessary, yet there is no method that has aimed to tackle this challenge in the TSAD field **(i)**.

Second, it is important to capture both intra-variable (aka temporal) and inter-variable (aka spatial) dependencies in MTS Ho et al. [2025]. However, existing studies are unable to effectively capture them. Regarding the intra-variable dependencies, they often preprocessed data by segmenting the signals into short time intervals Lai et al. [2024], or applied conventional networks such as recurrent neural networks or transformers Yang et al. [2023]. Such learned dependencies imply that observations close to each other are expected to

be similar. This is problematic as trends, seasonality and unpredictability are always present in MTS Ho et al. [2025]. Thus, developing a method that can handle the *long-range* intra-variable dependencies to aid in distinguishing normal and abnormal variations in MTS is crucial **(ii)**.

Regarding the inter-variable dependencies, very recently, graphs have brought the potential to model the relationships between variables (sensors) in MTS. By representing variables as nodes and their connections as edges, graphs provide an intuitive way to understand the underlying relationships between variables - a useful property in TSAD. For example, the changes in one variable can be used to predict the changes in another if they are correlated. However, modeling graphs to effectively capture this dependency type is challenging. Existing studies proposed to pre-define graphs based on prior knowledge, e.g., the known locations of sensors Tang et al. [2021], Ho and Armanfard [2023], Hojjati et al. [2023]. However, in many real-world applications, pre-defining the graph properties such as node locations and node features is not practical due to dynamic testing environments – e.g., electroencephalogram sensor locations in comatose/epilepsy patients may vary depending on the brain damaged regions. Hence, instead of pre-defining graphs, dynamically learning the graph over time is highly desirable.

Lastly, while long-range intra-variable and inter-variable dependencies are both important for TSAD, designing an effective joint learning framework to capture them is yet challenging. Recent studies have shown that there are two groups of a combined model: time-and-graph and time-then-graph Gao and Ribeiro [2022]. The time-and-graph approach first constructs graphs and then embeds a temporal network. On the other hand, the time-then-graph framework first projects time-series data to a temporal network, the extracted temporal features are then used to model graphs. It is shown that compared to the time-and-graph framework, the time-then-graph approach achieves a significant improvement in classification and regression tasks Gao and Ribeiro [2022], Tang et al. [2023]. Yet, till now, no study has explored the time-then-graph framework for unsupervised TSAD **(iii)**.

Based on the above observations we propose a novel approach, called TSAD-C, that addresses **(i)-(iii)** challenges. The Related Work section is provided in **Appendix A**. The main contributions of this paper are described as follows:

- We propose a novel *fully* unsupervised approach, namely TSAD-C, trained on contaminated data in an end-to-end manner to detect all types of anomalies in MTS. To the best of our knowledge, this is the first study that uses contaminated data in the training phase for TSAD, addressing a much more challenging problem than the existing studies.
- TSAD-C consists of three modules, namely Decontaminator, Long-range Variable Dependency Modeling, and Anomaly Scoring. The initial module aims to identify and eliminate abnormal patterns that are likely to be anomalies. This step results in decontaminated data, which is prepared swiftly for subsequent modules. The second module is a time-then-graph approach that is designed to model the long-range intra- and inter-variable dependencies within the decontaminated data. The last module computes anomaly scores to detect anomalies.
- The novel Decontaminator employs masking strategies and a structure state space (S4)-based conditional diffusion model, while the second module integrates Intra-variable Modeling, and Inter-variable Modeling components. The Anomaly Scoring module leverages insights of the first two modules.
- Extensive experiments on four reliable, diverse, and challenging datasets demonstrate that our method outperforms existing studies, thus establishing a new state-of-the-art in the TSAD field.

2 PROPOSED METHOD

A dataset is defined as $X = (\mathbf{x}_{(1)}, \mathbf{x}_{(2)}, \ldots, \mathbf{x}_{(N)})$, where $\mathbf{x}_{(i)} = (x^1_{(i)}, x^2_{(i)}, \ldots, x^K_{(i)})$ is the ith observation in the time series of N observations, $\mathbf{x}_{(i)} \in \mathbb{R}^{K \times L}$, K and L denote the number of variables (sensors) and the length of the ith observation, respectively. Our task is to detect anomalous observations from all types in the test data X_{test} by training the model with contaminated data X_{train}. No information about anomalies that contaminate normal data is provided during training, such as their labels or their positions within the time series. A validation set X_{valid} is used for early stopping and finding the decision threshold. The block diagram of the proposed TSAD-C method, depicting the three modules, is shown in Figure 1. Details of each module are presented below. The pseudocodes are provided in **Appendix B**. The source code is also available in the provided zip file.

2.1 DECONTAMINATOR

This module incorporates masking strategies and an S4-based conditional diffusion model. As no information about anomalies is provided during training, we propose masking strategies to decontaminate the input data. The diffusion model is then deployed to rectify anomalies, with S4 - a noise estimator included to ensure that long-range intra-variable dependencies are effectively captured. Notably, we introduce a pioneering concept in the diffusion field, i.e., minimizing the noise error on masked portions for a simpler and more streamlined training process. The decontaminated data is then obtained by a *single* step during the reverse process, which is a fast data preparation for subsequent modules – a significant advantage for practical applications.

Masking Strategies. Following the practical scenarios, we assume that normal samples significantly outnumber anoma-

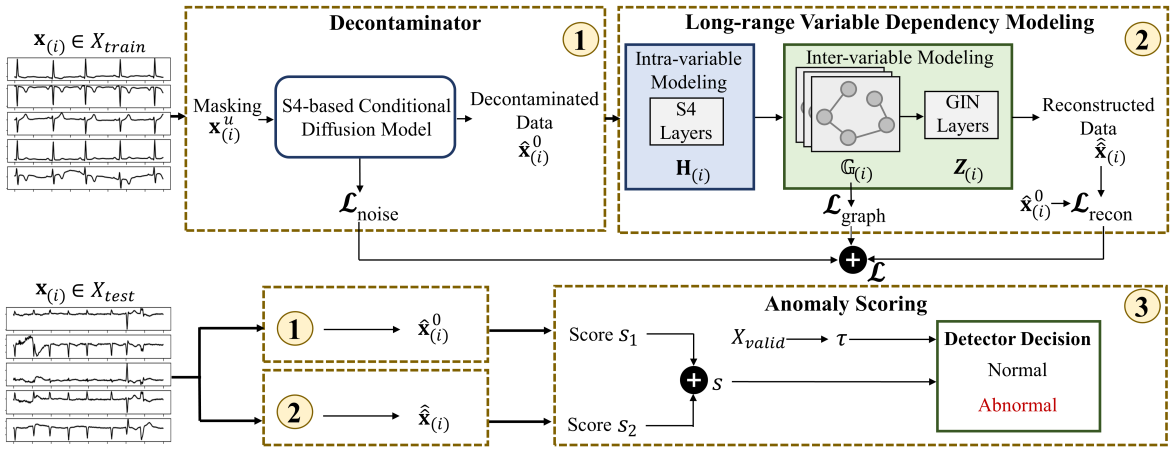

Figure 1: The overall framework of TSAD-C consists of three modules: the Decontaminator integrates masking strategies and an S4-based diffusion model, the Long-Range Variable Dependency Modeling module incorporates Intra- and Inter-variable Modeling components; and the Anomaly Scoring module leverages insights from the preceding modules to detect anomalies.

lies. When masking a portion of X_{train}, both normal and abnormal patterns might be removed. As normal data predominates in X_{train}, omitting some normal patterns is not likely to yield detrimental consequences as the substantial amount of remaining normal data can compensate for masked portions. Conversely, masking can help reducing the proportion of anomalies. This benefits the downstream module as it facilitates the learning process of variable dependencies that characterize the underlying behavior of normality.

We define a mask as $\mathbf{v} \in \{0, 1\}$, $\mathbf{v} \in \mathbb{R}^{K \times L}$, where zeros and ones denote the values to be masked and the values to be kept, respectively. Hence, the ith masked observation is $\mathbf{x}^u_{(i)} = \mathbf{x}_{(i)} \odot \mathbf{v}$, where $\mathbf{x}^u_{(i)} \in \mathbb{R}^{K \times L}$ and $\odot$ denotes point-wise multiplication. We perform three masking scenarios, namely random masking (RandM), random block masking (RandBM) and blackout masking (BoM) Alcaraz and Strodthoff [2022]. We control the masking ratio by the hyperparameter r, which specifies the number of timestamps to be masked. RandM randomly samples r to be masked across variables. In RandBM, there might be no time overlap between the masked windows across variables, whereas in BoM, the same time window is masked across all variables. Note that each masked window has the size of r.

S4-based Conditional Diffusion Model. This is developed based on a diffusion model Croitoru et al. [2023] that includes the diffusion and reserve processes. In this paper, the diffusion process incrementally adds Gaussian noise to the initial stage of $\mathbf{x}^u_{(i)}$, called $\mathbf{x}^0_{(i)}$, over T diffusion steps:

$$p(\mathbf{x}^1_{(i)}, \ldots, \mathbf{x}^T_{(i)} | \mathbf{x}^0_{(i)}) = \prod_{t=1}^{T} p(\mathbf{x}^t_{(i)} | \mathbf{x}^{t-1}_{(i)}), \tag{1}$$

where $p(\mathbf{x}^t_{(i)} | \mathbf{x}^{t-1}_{(i)}) := \mathcal{N}(\mathbf{x}^t_{(i)}; \mu^t_{(i)}, \sigma^t_{(i)})$. This indicates that $\mathbf{x}^t_{(i)}$ is sampled from a normal distribution with mean $\mu^t_{(i)} = \sqrt{1-\beta_t}\mathbf{x}^{t-1}_{(i)}$ and variance $\sigma^t_{(i)} = \beta_t \mathbf{I}$. $\mathbf{I}$ is the identity matrix, $\beta_t \in (0, 1)$ is a variance scheduler that controls the quantity of noise added at the tth diffusion step. In our implementation, we increase β_t linearly from 10^{-4} to 0.02. By setting $\alpha_t = 1 - \beta_t$, $\bar{\alpha}_t = \prod_{j=1}^{t} \alpha_j$, the diffusion process allows to immediately transform $\mathbf{x}^0_{(i)}$ to a noisy $\mathbf{x}^t_{(i)}$ according to β_t in a closed form as $\mathbf{x}^t_{(i)} = \sqrt{\bar{\alpha}_t}\mathbf{x}^0_{(i)} + \sqrt{1-\bar{\alpha}_t}\epsilon_t$ where the noise $\epsilon_t \sim \mathcal{N}(0, \mathbf{I})$, $\epsilon_t \in \mathbb{R}^{K \times L}$. We add noise to both masked and non-masked portions of $\mathbf{x}^0_{(i)}$. As the diffusion step increases, $\mathbf{x}^0_{(i)}$ gradually loses its distinguishable features and approaches a Gaussian distribution; hence, both anomalous and normal patterns appear indistinguishable.

The reverse process is parameterized by θ as:

$$q_\theta(\mathbf{x}^0_{(i)}, \ldots, \mathbf{x}^{T-1}_{(i)} | \mathbf{x}^T_{(i)}) = \prod_{t=1}^{T} q_\theta(\mathbf{x}^{t-1}_{(i)} | \mathbf{x}^t_{(i)}), \tag{2}$$

where each $q_\theta(\mathbf{x}^{t-1}_{(i)} | \mathbf{x}^t_{(i)}) := \mathcal{N}(\mathbf{x}^{t-1}_{(i)}; \mu_\theta(\mathbf{x}^t_{(i)}, t, c), \sigma_\theta(\mathbf{x}^t_{(i)}, t, c)^2\mathbf{I})$. μ_θ and σ_θ are parameterized as:

$$\begin{aligned} \mu_\theta(\mathbf{x}^t_{(i)}, t, c) &= \tfrac{1}{\sqrt{\alpha_t}}\Big(\mathbf{x}^t_{(i)} - \tfrac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon_\theta(\mathbf{x}^t_{(i)}, t, c)\Big), \\ \sigma_\theta(\mathbf{x}^t_{(i)}, t, c) &= \sqrt{\bar{\beta}_t}, \end{aligned} \tag{3}$$

where $\bar{\beta}_t = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$ and $\bar{\beta}_1 = \beta_1$. ϵ_θ is a noise estimator, which takes $\mathbf{x}^t_{(i)}$, the diffusion step t and a conditional factor c as the inputs and aims to predict the noise from $\mathbf{x}^t_{(i)}$. c is a concatenation of non-masked parts in $\mathbf{x}^u_{(i)}$ and the positional information of masked parts provided by $\mathbf{v}$. This extra information facilitates our reverse process to distinguish the zero portions of non-masked and masked parts.

Note that ϵ_θ plays a key role in our reverse process. Since capturing long-range intra-variable dependencies is crucial, we propose to build ϵ_θ based on S4 Gu et al. [2022] - a recent deep sequence model with the concept of a state

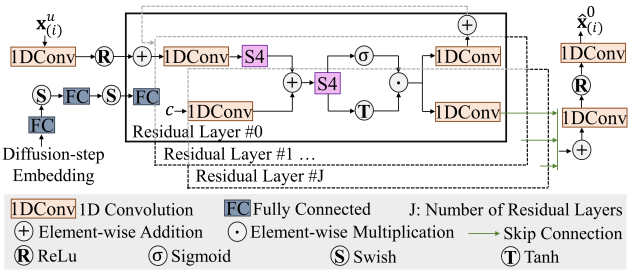

Figure 2: The architecture of the Decontaminator includes two S4 layers in every residual block to ensure that long-range intra-variable dependencies are effectively captured.

space model (SSM). A continuous-time SSM maps $\mathbf{x}^t_{(i)}$ to a high dimensional state $h^t_{(i)}$ before projecting it to the output $\mathbf{y}^t_{(i)}$. This transition can be defined as:

$$\tilde{h}^t_{(i)} = Ah^t_{(i)} + B\mathbf{x}^t_{(i)} \text{ and } \mathbf{y}^t_{(i)} = Ch^t_{(i)} + D\mathbf{x}^t_{(i)}, \quad (4)$$

where A, B, C, D are transition matrices learned by gradient descent. However, S4 shows that a discrete-time SSM can be represented as a convolution operation by:

$$\overline{O} := (\overline{CB}, \overline{CAB}, \ldots, \overline{CA}^{L-1}\overline{B}), \mathbf{y}_{(i)} = \overline{O} * \mathbf{x}_{(i)}, \quad (5)$$

where $\overline{A}, \overline{B}, \overline{C}$ are the discretized matrices, $\overline{CA}^{L-1}$ denotes the multiplication of discretized matrices at $L-1$, and $\overline{O}$ is a SSM convolution kernel. D is omitted in Equation (5) as $D\mathbf{x}^t_{(i)}$ can be viewed as a skip connection. S4 parameterizes A as a diagonal plus low rank matrix that enables fast computation of $\overline{O}$. It also includes the HiPPO matrices Gu et al. [2020] capable of capturing long-term intra-variable dependencies. We employ two S4 layers, one after the addition of the embeddings related to $\mathbf{x}^u_{(i)}$ and another layer after including c in the residual blocks, shown in Figure 2.

To have a simpler and more streamlined reverse process during the training of ϵ_θ, we suggest minimizing the noise error on the masked parts shown in Equation (6). Note that the masked parts only consist of noise without any actual data patterns, unlike the non-masked parts containing both noise and the actual data patterns. Hence, deriving noise estimated from the masked data is a more straightforward task and can be accomplished using less intricate networks. This approach also speeds up data preparation for subsequent modules. We provide **Section 3.2.5** that compares two optimization strategies: minimizing $\mathcal{L}_{\text{noise}}$ on the masked portions versus minimizing $\mathcal{L}_{\text{noise}}$ on the entire observations as done by prior diffusion studies Alcaraz and Strodthoff [2022], Chen et al. [2024]. It shows that the former approach yields superior performance, simplicity and applicability.

$$\mathcal{L}_{\text{noise}} = \|\epsilon_t \odot (1-\mathbf{v}) - \hat{\epsilon}_t \odot (1-\mathbf{v})\|^2, \quad (6)$$

where $\hat{\epsilon}_t$ is the predicted noise obtained from $\epsilon_\theta(\mathbf{x}^t_{(i)}, t, c)$.

We then obtain the decontaminated data in the training as:

$$\hat{\mathbf{x}}^0_{(i)} = \frac{1}{\sqrt{\bar{\alpha}_T}}\left(\mathbf{x}^T_{(i)} - \sqrt{1-\bar{\alpha}_T}\hat{\epsilon}_T\right). \quad (7)$$

Note that approximating $\hat{\mathbf{x}}^0_{(i)}$ by a *single* step (immediately at the Tth step) enables a faster reverse speed as $\hat{\mathbf{x}}^0_{(i)}$ serves as the input for the second module during training. Meanwhile, during testing, we perform a complete sampling step from T to 1 based on Equations (2) and (3) to obtain $\hat{\mathbf{x}}^0_{(i)}$.

Theoretical Analysis of Decontamination Effectiveness. Reducing the fraction of anomalies (*"impurity"*) in the training data is crucial for accurate anomaly detection. Training on less contaminated data helps the model learn normal patterns more effectively, thereby improving its ability to detect genuine anomalies.

We define N as the total number of training samples, N_a as the number of truly anomalous samples, $\eta_a = \frac{N_a}{N}$ as the *true* anomaly ratio (with $0 < \eta_a < 0.5$), $\eta_e \in (0,1)$ as the *estimated* anomaly ratio chosen by the user, and $ip_1 = \eta_a$ as the initial impurity level.

Impurity Reduction via Decontaminator Masking. During training, the Decontaminator masks a fraction η_e of all samples *uniformly at random*. Hence, on average, $\eta_e N_a$ anomalous samples are also masked. Since $0 < \eta_a < 0.5$, the new impurity level after decontamination is:

$$ip_2 = \frac{N_a - \eta_e N_a}{N} = \eta_a\,(1-\eta_e).$$

Since $0 < \eta_e < 1$, we have

$$ip_2 = \eta_a\,(1-\eta_e) < \eta_a = ip_1.$$

Thus, the impurity always *strictly decreases* compared to the original contamination level.

User-Controlled Impurity Reduction. Users can *control* how much to reduce the impurity by selecting η_e. Suppose one wishes to reduce the original impurity $ip_1 = \eta_a$ to a fraction $\alpha \in (0,1)$:

$$ip_2 = \alpha\, ip_1 = \alpha\,\eta_a.$$

Solving $\eta_a(1-\eta_e) = \alpha\,\eta_a$ yields:

$$\eta_e = 1 - \alpha.$$

For instance, if $\alpha = 0.5$, then one sets $\eta_e = 0.5$ to remove 50% of the initial impurity.

Robustness to Underestimation and Overestimation of η_a. In practice, η_a is unknown. Nevertheless, the Decontaminator remains robust whether η_e is an underestimate or overestimate.

Regarding underestimation ($\eta_e < \eta_a$), we still have $ip_2 = \eta_a(1-\eta_e) < \eta_a = ip_1$. More specifically,

$$ip_1 - ip_1^2 < ip_2 < ip_1.$$

Hence, even if one underestimates the contamination level, the Decontaminator still *strictly* reduces impurity.

Regarding overestimation ($\eta_e \geq \eta_a$), we have

$$ip_2 = \eta_a(1-\eta_e) \leq \eta_a(1-\eta_a) = ip_1 - ip_1^2 \leq ip_1.$$

Since $\eta_a < 0.5$, the maximum of $ip_1 - ip_1^2$ over $\eta_a \in (0, 0.5)$ is 0.25. Thus, $ip_2 \leq 0.25$. Even when a large fraction of samples is masked (e.g., $\eta_e = 0.7$), as long as sufficient normal data remains, the model can still learn normal structure effectively.

Conclusion. Regardless of how η_e relates to η_a, the Decontaminator *strictly reduces* the contamination level in the training set ($ip_2 < ip_1$). This guarantees a cleaner dataset, where the model's representations are primarily driven by normal patterns. As a result, reconstruction-based anomaly detection becomes more robust, even when η_a is inaccurately estimated.

2.2 LONG-RANGE VARIABLE DEPENDENCY MODELING

This module builds a time-then-graph framework, motivated in **Section 1**-Paragraph 5, by incorporating two components: Intra-variable Modeling and Inter-variable Modeling.

Intra-variable Modeling. We propose to leverage multiple back-to-back S4 layers to capture long-range intra-variable dependencies. Specifically, we use $\hat{\mathbf{x}}^0_{(i)} \in \mathbb{R}^{K \times L}$ as the input and project it onto an embedding space, called $\mathbf{H}_{(i)} \in \mathbb{R}^{K \times \Gamma \times U}$, where Γ and U are hyperparameters defining the S4 embedding dimension. To maintain a sense of the number of timestamps present in $\mathbf{x}_{(i)}$, we set Γ to L, corresponding to the input length. This allows us to model long-term intra-variable dependencies within each variable. $\mathbf{H}_{(i)}$ is then used in the graph learning phase to model inter-variable dependencies. Prior studies have shown the superior performance of graph learning when using the temporal embedding rather than the original data Tang et al. [2023].

Inter-variable Modeling. We represent $\mathbf{H}_{(i)}$ as a set of graphs $\mathbb{G}_{(i)} = \{\mathcal{G}^m_{(i)}\}^d_{m=1}$, where $d = \frac{\Gamma}{g}$ and g is the predefined length of the short and non-overlapping time windows within the ith observation. Since each observation can encompass thousands of timestamps, constructing a graph for every time step becomes inefficient and computationally demanding. Hence, we create a graph over a defined time window, which aids in information aggregation. This strategy not only leads to a graph with reduced noise but also facilitates faster computations Gao and Ribeiro [2022].

We define $\mathcal{G}^m_{(i)} = \{\mathbf{E}^m_{(i)}, \mathcal{A}^m_{(i)}\}$. $\mathbf{E}^m_{(i)} \in \mathbb{R}^{K \times U}$ denotes the embedding derived by averaging the elements of $\mathbf{H}_{(i)}$ along its second dimension. $\mathcal{A}^m_{(i)} \in \mathbb{R}^{K \times K}$ is the adjacency matrix. Each row and column in $\mathcal{A}^m_{(i)}$ correspond to a node (variable). The non-zero value indicates that there exists an edge connecting the two nodes. We then employ a self-attention paradigm Tang et al. [2023] in which attention weights are assigned to the edges' weights, represented as: $\mathbf{Q} = \mathbf{E}^m_{(i)}\mathbf{W}^{\mathbf{Q}}, \mathbf{R} = \mathbf{E}^m_{(i)}\mathbf{W}^{\mathbf{R}}, \mathcal{A}^m_{(i)} = \text{softmax}(\frac{\mathbf{Q}\mathbf{R}^\top}{\sqrt{D}})$, where $\mathbf{W}^{\mathbf{Q}}, \mathbf{W}^{\mathbf{R}} \in \mathbb{R}^{U \times U}$ are the learnable weights that project $\mathbf{E}^m_{(i)}$ to the query $\mathbf{Q}$ and the key $\mathbf{R}$, respectively.

To help guiding the graph learning process, we also include a pre-defined adjacency matrix, called $\mathcal{A}'^m_{(i)}$, based δ-nearest neighbors. Its edge values are computed by the cosine similarity between the nodes' embeddings in $\mathbf{E}^m_{(i)}$. We keep the top δ edges that have the highest values for each node to avoid overly connected graphs. In our experiments, $\delta = 3$. Hence, the final $\mathcal{A}^m_{(i)} = \zeta\mathcal{A}'^m_{(i)} + (1-\zeta)\mathcal{A}^m_{(i)}$, where the hyperparameter $\zeta \in [0, 1)$ balances the two components.

It is important to regularize the graph to ensure desired graph properties such as smoothness (the features should change smoothly between neighboring nodes), sparsity (avoiding an overly connected graph) and connectivity (avoiding a disconnected graph) Zhu et al. [2022]. Hence, we include three constraints in the regularization loss as:

$$\begin{aligned}\mathcal{L}_{\text{graph}} = \frac{1}{d}\sum_{m=1}^{d} \xi_1\mathcal{L}_{\text{smooth}}(\mathbf{E}^m_{(i)}, \mathcal{A}^m_{(i)}) + \\ \xi_2\mathcal{L}_{\text{sparse}}(\mathcal{A}^m_{(i)}) + \xi_3\mathcal{L}_{\text{connect}}(\mathcal{A}^m_{(i)}),\end{aligned} \tag{8}$$

where $\mathcal{L}_{\text{smooth}} = \frac{1}{K^2}\text{tr}(\mathbf{E}^{m\top}_{(i)} M_{\text{Lap}}\mathbf{E}^m_{(i)})$, $M_{\text{Lap}} = M_{\text{degree}} - \mathcal{A}^m_{(i)}$ is the Laplacian matrix, M_{degree} is the degree matrix of $\mathcal{A}^m_{(i)}$, and $\text{tr}(\cdot)$ denotes the trace. $\mathcal{L}_{\text{sparse}} = \frac{1}{K^2}\|\mathcal{A}^m_{(i)}\|^2_F$ and $\|\cdot\|_F$ is the Frobenius norm. $\mathcal{L}_{\text{connect}} = -\frac{1}{K}\mathbf{1}^\top \log(\mathcal{A}^m_{(i)} \cdot \mathbf{1})$, and $\mathbf{1} \in \mathbb{R}^{K \times 1}$ is a matrix of ones. ξ_1, ξ_2 and ξ_3 are hyperparameters defined to balance the terms in $\mathcal{L}_{\text{graph}}$.

We then leverage a graph isomorphism network (GIN), which shows a strong representational power Xu et al. [2019] to capture inter-variable dependencies between nodes in $\mathcal{G}^m_{(i)}$. The embedding of nodes in $\mathcal{G}^m_{(i)}$ is represented as $\mathbf{z}^m_{(i)} := \text{GIN}(\mathbf{E}^m_{(i)}, \mathcal{A}^m_{(i)})$, $\mathbf{z}^m_{(i)} \in \mathbb{R}^{K \times g \times U}$. We concatenate the node embeddings of all graphs within $\mathbb{G}_{(i)}$ as $\mathbf{Z}_{(i)} = \text{concat}(\mathbf{z}^1_{(i)}, \ldots, \mathbf{z}^d_{(i)})$. Finally, a linear layer is added to obtain reconstructed data $\hat{\hat{\mathbf{x}}}_{(i)} := \text{Linear}(\mathbf{Z}_{(i)})$, $\hat{\hat{\mathbf{x}}}_{(i)} \in \mathbb{R}^{K \times L}$. Thus, the reconstruction loss is denoted as:

$$\mathcal{L}_{\text{recon}} = \|\hat{\mathbf{x}}^0_{(i)} - \hat{\hat{\mathbf{x}}}_{(i)}\|^2. \tag{9}$$

The final loss in the training phase is defined as :

$$\mathcal{L} = \mathcal{L}_{\text{noise}} + \mathcal{L}_{\text{graph}} + \mathcal{L}_{\text{recon}}. \tag{10}$$

2.3 ANOMALY SCORING

In the test phase, we compute an anomaly score based the root mean square error (RMSE) for each $\mathbf{x}_{(i)}$. Specifically,

we project $\mathbf{x}_{(i)}$ to the Decontaminator where masking strategies and the complete sampling step from T to 1 are applied to obtain $\hat{\mathbf{x}}^0_{(i)}$. If $\mathbf{x}_{(i)}$ is an anomaly, the masked portions are expected to be inaccurately sampled. Additionally, instead of using $\hat{\mathbf{x}}^0_{(i)}$ as the input for the second module as done during training, $\mathbf{x}_{(i)}$ is directly used to obtain $\hat{\hat{\mathbf{x}}}_{(i)}$ in the test phase. The assumption is that if $\mathbf{x}_{(i)}$ is an anomaly, the second module with the goal of achieving the flawless reconstruction of normal patterns would be unable to reconstruct it. The final score s for each $\mathbf{x}_{(i)}$ is computed as:

$$\begin{aligned} s_1 &= \left(\frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{K}\left((\hat{\mathbf{x}}^0_{(i)} - \mathbf{x}_{(i)}) \odot (1-\mathbf{v})\right)^2\right)^{0.5}, \\ s_2 &= \left(\frac{1}{L}\sum_{l=1}^{L}\sum_{k=1}^{K}\left(\hat{\hat{\mathbf{x}}}_{(i)} - \mathbf{x}_{(i)}\right)^2\right)^{0.5}, \\ s &= \lambda_1 s_1 + \lambda_2 s_2, \end{aligned} \tag{11}$$

where s_1 and s_2 are, respectively, the RMSE scores obtained from the first and second modules. λ_1 and λ_2 are hyperparameters defined to ensure that scores from both modules fall within a similar numeric range. For a fair comparison, they are fixed across all experiments and datasets.

We conduct a decision threshold search on the *unlabeled* X_{valid}, deviating from the TSAD practice where the threshold is selected on a *labeled* set Carmona et al. [2022], Chen et al. [2022]. This approach is impractical for unsupervised applications where labeled data is unavailable. This may also lead to overfitting when labeled data is not sufficient to represent the distribution of anomalies, a common challenge in TSAD. In our case, we determine the threshold τ by a quantile approach Kuan et al. [2017] applied to anomaly scores obtained from the unlabeled X_{valid}. Specifically, we select a quantile based on a rough estimation of the percentage of normal data in X_{valid}, as provided by the dataset provider. For instance, if about 20% of the data is contaminated, we set the quantile to 80%. In the test phase, observations with s above τ are detected as anomalies. Note that there is no overlap between X_{train}, X_{valid}, and X_{test}.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETTINGS

This section introduces the datasets, baselines and evaluation metrics in our study. Implementation details are provided in **Appendix E**. Due to the diversity in dataset characteristics and dimensions, we present dedicated discussions on TSAD-C's computational cost and scalability in **Appendix G**.

Datasets. Many TSAD methods have relied on benchmark datasets such as Yahoo, NASA, SWaT, WADI, SMAP, and MSL. However, these datasets are unreliable due to (i) mislabeled ground truth, (ii) triviality, (iii) unrealistic anomaly density and (iv) run-to-failure bias Wu and Keogh [2021]. This renders them unsuitable for evaluating TSAD methods.

Table 1: The numbers of observations are shown in X_{train}, X_{valid} and X_{test} of each dataset, with the anomaly ratio η in parentheses. f_s is the sampling rate (in Hertz), K is the number of variables, and L is the length of an observation.

Dataset	X_{train} (η)	X_{valid} (η)	X_{test} (η)	f_s	K	L
SMD	1624 (10.3%)	276 (7.97%)	416 (33.8%)	$\frac{1}{60}$	38	600
ICBEB	910 (20.0%)	82 (20.7%)	222 (59.9%)	100	12	6,000
DODH	2515 (19.8%)	320 (21.8%)	310 (51.6%)	250	16	7,500
TUSZ	5275 (17.0%)	1055 (20.0%)	1581 (40.0%)	200	19	12,000

Aware of these issues, we carefully select four datasets that are reliable, diverse, yet challenging. They are recorded using various sensor systems, each varying in the number and types of sensors, leading to distinct characteristics. These include **SMD** Su et al. [2019], **ICBEB** Liu et al. [2018], **DODH** Guillot et al. [2020], and **TUSZ** Shah et al. [2018].

Specifically, SMD is an industrial dataset consisting of five weeks of data from 28 server machines and widely used in the TSAD field. While not perfect, SMD is considered of much higher quality compared to criticized benchmarks Wagner et al. [2023] (see **Appendix C.1**). ICBEB, DODH and TUSZ are well-established yet challenging datasets from the biomedical domain, have not received criticisms (i)-(iv), and importantly, they reflect real-world scenarios with diverse anomaly types, while other datasets such as SMD contain only one anomaly type. ICBEB is an ECG database, consisting of normal heart rhythms and five anomaly types. DODH is a sleep database, with the deepest sleep stage as normal and two anomaly sleep types. TUSZ is an EEG database, with normal resting-state brain activities and two anomaly seizure types. Statistics of each dataset are shown in Table 1, with further details in **Appendix C**. Note that we include 10-20% anomalies of all types in X_{train} to contaminate normal data during training, increasing the challenge for algorithms to detect all anomaly types in the test phase.

Baselines. We compare TSAD-C against 12 SOTA unsupervised methods from the TSAD literature, ranging from autoencoders, self-supervised, transformers to diffusion approaches. For a fair comparison, we do not include methods that require transfer learning with any additional datasets. We categorize all methods into three groups based on their ability to capture either Intra-, Inter- or Both-variable dependencies. Specifically, Intra- methods include USAD Audibert et al. [2020], LSTM-AE Wei et al. [2023], S4-AE Gu et al. [2022] and DCdetector Yang et al. [2023]. Inter-methods are GAE Du et al. [2022], GDN Deng and Hooi [2021], and EEG-CGS Ho and Armanfard [2023]. Methods addressing Both- include InterFusion Li et al. [2021], DVGCRN Chen et al. [2022], GRU-GNN and GraphS4mer Tang et al. [2023], IMDiffusion Chen et al. [2024] and our method. Details of the baselines are shown in **Appendix D**.

Evaluation Metrics. We employ F1-score (F1), Recall

Table 2: Comparison of existing methods and TSAD-C. The best and second-best scores are in bold and underlined.

Group	Method	SMD			ICBEB			DODH			TUSZ			Average
		F1	Rec	APR	F1	Rec	APR	F1	Rec	APR	F1	Rec	APR	F1
Intra-	USAD	0.261	0.227	0.398	0.579	0.485	0.705	0.355	0.419	0.514	0.450	0.393	0.581	0.411
	LSTM-AE	0.332	0.411	0.445	0.609	0.651	0.752	0.534	0.706	0.643	0.471	0.424	0.592	0.486
	S4-AE	0.313	0.305	0.432	0.664	0.735	0.749	0.625	0.821	0.713	0.527	0.576	0.615	0.532
	DCdetector	0.318	0.276	0.448	0.626	0.598	0.718	0.442	0.550	0.576	0.485	0.447	0.600	0.468
Inter-	GAE	0.241	0.191	0.395	0.575	0.454	0.746	0.480	0.613	0.604	0.508	0.476	0.625	0.451
	GDN	0.301	0.283	0.423	0.586	0.515	0.741	0.524	0.687	0.636	0.456	0.398	0.585	0.467
	EEG-CGS	0.295	0.291	0.415	0.561	0.470	0.740	0.502	0.650	0.620	0.516	0.490	0.619	0.469
Both-	InterFusion	0.383	0.504	0.490	0.649	0.651	0.753	0.418	0.512	0.559	0.532	0.520	0.628	0.496
	GRU-GNN	0.329	0.383	0.440	0.647	0.689	0.742	0.587	0.806	0.684	0.506	0.474	0.613	0.517
	DVGCRN	0.323	0.305	0.442	0.615	0.575	0.744	0.480	0.612	0.604	0.397	0.322	0.554	0.479
	GraphS4mer	0.405	0.567	0.514	0.638	0.667	0.738	0.565	0.762	0.667	0.524	0.511	0.621	0.533
	IMDiffusion	0.426	0.603	0.533	0.611	0.553	0.750	0.544	0.725	0.651	0.381	0.452	0.532	0.491
	TSAD-C	**0.479**	**0.801**	**0.604**	**0.707**	**0.841**	**0.773**	**0.652**	**0.843**	**0.728**	**0.545**	**0.830**	**0.652**	**0.596**

(Rec), and Area Under the Precision-Recall Curve (APR) to comprehensively assess the performance of each method.

3.2 EXPERIMENTAL RESULTS

3.2.1 Comparison with State-of-the-Art

The performances of all methods are presented in Table 2, where TSAD-C uses RandBM. It shows that TSAD-C surpasses all existing studies and achieves an average improvement of 6.3% in F1 compared to the second-best method. TSAD-C also obtains a significant improvement in Rec – the ability to correctly detect most of anomalies. This improvement is crucial when dealing with contaminated data, where existing methods failed to detect the anomaly types similar to those encountered during training due to their assumption of clean training data, leading to misdetection of such anomalies. Additionally, existing methods handling Both- generally outperform those addressing only one dependency type, supporting our assumption that both dependency types are crucial. For example, GraphS4mer outperforms models that focus on a single dependency type, such as USAD, which addresses only temporal aspects, or GAE, which focuses on spatial dependencies. Moreover, methods handling long-range Intra- (e.g., S4-AE) outperform those concentrating solely on Inter-. This suggests that while inter-variable dependencies are also important, they often represent variable relationships that can be captured over shorter temporal windows, which might miss the broader temporal context necessary to detect anomalies effectively.

3.2.2 Resilience to Contamination Levels

This section verifies TSAD-C's performance through two additional experiments on ICBEB: (1) varying the number of anomaly types and (2) varying the anomaly ratio η in X_{train} and X_{valid}. We also select five methods, each achieving high performance within its category, for comparison.

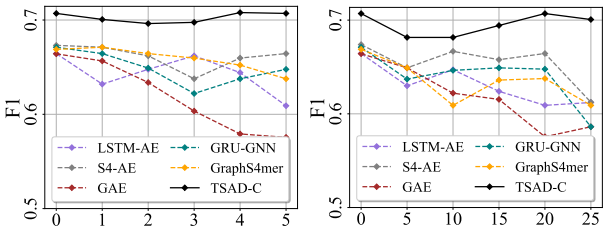

Figure 3: (**Left**) F1 score versus the number of anomaly types κ. (**Right**) F1 score versus the anomaly ratio η.

Variability on The Anomaly Types. We assess TSAD-C's performance in detecting five anomaly types available in X_{test} while not all types are present in X_{train} and X_{valid}. We introduce κ as the number of anomaly types – e.g., $\kappa = 2$ signifies two anomaly types present in X_{train} and X_{valid}. Note that η remains constant. Figure 3 (**Left**) shows that the performance of existing methods diminishes as κ increases since they all assume the input data as pure normal data, hence are incapable of handling contaminated data. Remarkably, TSAD-C consistently attains the highest F1, irrespective of changes in κ. This underlines our method's denoising prowess as it remains effective regardless of the diversity in available anomaly types.

Variability on The Anomaly Ratio. We investigate the robustness of TSAD-C by varying η. Note that all anomaly types ($\kappa = 5$) are present within each subset of the dataset for this experiment. Figure 3 (**Right**) shows the consistent performance of TSAD-C across different anomaly ratios, demonstrating the Decontaminator's effectiveness in TSAD-C. Meanwhile, the performance of other unsupervised methods tends to decline as normal data impurity increases.

3.2.3 Visualization of Normal Approximation

Figure 4 shows a visualization comparing the ground truth $\mathbf{x}_{(i)}$, decontaminated data $\hat{\mathbf{x}}^0_{(i)}$, and reconstructed data $\hat{\hat{\mathbf{x}}}_{(i)}$ of a masked segment of normal and abnormal samples in DODH. Visualizations on other datasets are shown in **Appendix F**. The segment of $\mathbf{x}_{(i)}$ is masked to zeros using BoM, before being processed by the Decontaminator. It is shown that $\hat{\mathbf{x}}^0_{(i)}$ and $\hat{\hat{\mathbf{x}}}_{(i)}$ fit $\mathbf{x}_{(i)}$ very well in the normal case, leading to a lower s. Meanwhile, there are significant fluctuations of $\hat{\mathbf{x}}^0_{(i)}$ and $\hat{\hat{\mathbf{x}}}_{(i)}$ compared to $\mathbf{x}_{(i)}$ in the abnormal case, resulting in a much higher s. This shows TSAD-C's effectiveness in distinguishing anomalies from normal data.

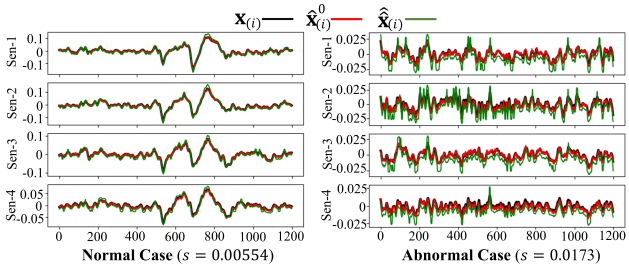

Figure 4: Comparison between normal and abnormal cases for the masked segment in DODH. Sen-k denotes the kth sensor. Each case includes $\mathbf{x}_{(i)}$, $\hat{\mathbf{x}}^0_{(i)}$ and $\hat{\hat{\mathbf{x}}}_{(i)}$.

3.2.4 Ablation Study

Certainly, handling contaminated data requires the indispensable inclusion of the Decontaminator. Excluding it would align the TSAD-C's detection results with those of other unsupervised methods. In this section, we aim to conduct ablation studies focusing on all three modules, i.e., Module (1) - Decontaminator, Module (2) - Long-range Variable Dependency Modeling, and Module (3) - Anomaly Scoring, as shown in Table 3. RandBM is used in all experiments. The sign "–" denotes the exclusion of a component.

Module (1). As stated earlier, the Decontaminator is fundamental to TSAD-C's ability to operate under the realistic setting of contaminated training data. Without it, the TSAD-C's detection results would align with those of other unsupervised anomaly detectors that assume clean training data, thereby undermining the key contribution of our work. To empirically demonstrate this, we evaluate TSAD-C with the Decontaminator removed, as shown in the first row of the table. The results demonstrate a clear performance drop, reaffirming our central claim that explicitly addressing contamination in the training data is critical for achieving robust and generalizable anomaly detection.

Module (2). The results are shown in the first and second rows, where Intra and Inter, respectively, denote the Intra-, and Inter-variable Modeling. Note that s_2 is computed by the non-removed block of Module (2), i.e., for the first and second rows, s_2 is determined by Inter and Intra, respectively. The results highlight the superior performance achieved by capturing long-range intra-variable dependencies, showing that while Inter is also an important component, having Intra is often prioritized due to the nature of time-series data.

Table 3: The performance of individual components in TSAD-C and their combinations. M. (1), M. (2) and M. (3) denote Module (1), Module (2) and Module (3), respectively.

TSAD-C		ICBEB			DODH			TUSZ		
		F1	Rec	APR	F1	Rec	APR	F1	Rec	APR
	– M. (1)	0.649	0.750	0.735	0.573	0.675	0.670	0.489	0.611	0.587
M. (2)	– Intra	0.651	0.735	0.738	0.565	0.637	0.667	0.499	0.655	0.598
	– Inter	0.686	0.803	0.759	0.622	0.762	0.705	0.535	0.789	0.639
M. (3)	– s_1	0.706	**0.856**	0.771	0.637	0.80	0.716	0.517	0.745	0.622
	– s_2	0.699	0.818	0.768	0.627	0.787	0.709	0.518	0.712	0.617
	All	**0.707**	0.841	**0.773**	**0.652**	**0.843**	**0.728**	**0.545**	**0.830**	**0.652**

Module (3). In this study, all components of Modules (1) and (2) are available during training, whereas we exclude either s_1 or s_2, respectively, obtained from Module (1) or Module (2) during testing. The results, shown in the third and fourth rows, indicate that by removing either s_1 or s_2, we observe a small drop in performance. This is due to the fact that in the test phase, we still perform data decontamination using the Decontaminator trained during training. Such data decontamination during testing resembles a case where one performs the unsupervised anomaly detection of the clean test data. This experiment can be considered as a demonstration of the Decontaminator's effectiveness. Importantly, integrating all components together (aka **All**) yields the best performance across all datasets, showcasing the synergistic complementarity of each component in enhancing the model's ability to detect anomalies.

3.2.5 Decontaminator Efficiency Study

As described in Section 2.1, we estimate the noise exclusively on the masked portions using $\mathcal{L}_{\text{noise}}$, shown in Equation (6). This approach simplifies and streamlines the reverse process and can be accomplished using less intricate networks. To empirically demonstrate this, we conduct additional experiments, where the noise is minimized over the entire observation (rather than just the masked parts), as commonly done in diffusion studies Chen et al. [2024]. This loss is defined as $\mathcal{L}^{\text{entire}}_{\text{noise}} = \|\epsilon_t - \hat{\epsilon}_t\|^2$. The results, shown in Table 4, indicate that minimizing $\mathcal{L}_{\text{noise}}$ exclusively on the masked portions achieves better performance, and is more applicable as the training time, is much faster, hence the decontaminated data can be quickly prepared for the second

Table 4: Comparison of minimizing $\mathcal{L}_{\text{noise}}^{\text{entire}}$ versus $\mathcal{L}_{\text{noise}}$. All components of TSAD-C are included in the training phase, whereas in the test phase, either s_1 (i.e., $-s_2$), s_2 (i.e., $-s_1$), or both s_1 and s_2 (i.e., $s_1 + s_2$) are included (min: minutes).

Dataset	TSAD-C	$\mathcal{L}_{\text{noise}}^{\text{entire}}$			$\mathcal{L}_{\text{noise}}$		
		Training time per epoch (min)	F1	APR	Training time per epoch (min)	F1	APR
ICBEB	$-s_1$	14.6	0.590	0.709	12.2	0.706	0.771
	$-s_2$	14.6	0.686	0.759	12.2	0.699	0.768
	$s_1 + s_2$	14.6	0.679	0.756	12.2	**0.707**	**0.773**
DODH	$-s_1$	21.8	0.617	0.702	19.3	0.637	0.716
	$-s_2$	21.8	0.600	0.703	19.3	0.627	0.709
	$s_1 + s_2$	21.8	0.623	0.700	19.3	**0.652**	**0.728**
TUSZ	$-s_1$	40.5	0.498	0.597	38.4	0.517	0.622
	$-s_2$	40.5	0.481	0.580	38.4	0.518	0.617
	$s_1 + s_2$	40.5	0.505	0.603	38.4	**0.545**	**0.652**

module. Meanwhile, minimizing $\mathcal{L}_{\text{noise}}^{\text{entire}}$ achieves suboptimal performance. We observe that this could be improved by increasing the complexity of the network, e.g., increasing the number of the network layers and the number of reverse steps. However, it would result in the slower training speed, which is not desirable for the Decontaminator in our task.

3.2.6 Effect of Masking Strategy

This section evaluates the performance of TSAD-C with various masking strategies. Note that we maintain a fixed value for r, which is used to control the masking ratio, to avoid hyperparameter fine-tuning. The consistency of better performance of TSAD-C even with a fixed r, irrespective of changes in the anomaly ratio η, is demonstrated in Figure 3 (**Right**). Table 5 shows that the performance of TSAD-C remains largely consistent when employing either RandM, RandBM, or BoM, across varying κ present in X_{train} and X_{valid}. Despite the consistency, RandBM showcases the most optimal performance. This can be attributed to introducing randomness into the masked time windows, which increases the level of robustness and adaptability to diverse patterns in the data. This is crucial for real-world applications where anomalies can manifest in diverse ways across sensors. RandBM emulates this diversity, enabling the model to learn and detect sensor-specific anomalies more effectively, thus boosting its overall performance.

4 CONCLUSION

This paper introduces TSAD-C, the first method trained on contaminated data to detect all types of anomalies in multivariate time series. TSAD-C comprises the Decontaminator, aimed at removing the potential anomalies during training and swiftly preparing decontaminated data for subsequent modules. The Long-range Variable Dependency Modeling

Table 5: Comparison of masking strategies employed in TSAD-C across varying numbers of anomaly types κ in X_{train} and X_{valid}. SD stands for standard deviation.

Dataset	Anomaly Type (κ)	RandM		RandBM		BoM	
		F1	APR	F1	APR	F1	APR
ICBEB	0	0.702	0.771	0.707	0.773	0.694	0.764
	1	0.669	0.748	0.687	0.759	0.671	0.752
	2	0.701	0.768	0.696	0.742	0.661	0.743
	3	0.681	0.755	0.697	0.751	0.675	0.751
	4	0.688	0.749	0.708	0.759	0.662	0.745
	5	0.694	0.764	0.707	0.773	0.669	0.747
	Mean ± SD	0.689 ± 0.012	0.759 ± 0.009	**0.700** ± 0.007	**0.760** ± 0.011	0.672 ± 0.011	0.750 ± 0.007
DODH	0	0.662	0.735	0.632	0.713	0.643	0.722
	1	0.637	0.717	0.671	0.742	0.618	0.703
	2	0.647	0.724	0.652	0.728	0.623	0.707
	Mean ± SD	0.649 ± 0.010	0.725 ± 0.007	**0.651** ± 0.016	**0.727** ± 0.012	0.628 ± 0.011	0.711 ± 0.008
TUSZ	0	0.541	0.649	0.530	0.639	0.531	0.640
	1	0.527	0.632	0.543	0.645	0.520	0.623
	2	0.519	0.631	0.545	0.652	0.531	0.640
	Mean ± SD	0.529 ± 0.009	0.637 ± 0.008	**0.539** ± 0.006	**0.645** ± 0.005	0.527 ± 0.005	0.634 ± 0.008

module is designed to capture long-range intra- and inter-variable dependencies, and provide an approximation of purified data. The Anomaly Scoring module integrates the capability of the first two modules. We demonstrate the superior performance of TSAD-C on four reliable, diverse and challenging datasets compared to existing methods.

ACKNOWLEDGMENTS

The authors would like to acknowledge the financial support of the Natural Sciences, Engineering Research Council of Canada (NSERC), Fonds de recherche du Quebec, and the Department of Electrical and Computer Engineering at McGill University. The authors also wish to thank Calcul Quebec and Compute Canada for providing the necessary computational resources to conduct our experiments.

References

Chuadhry Mujeeb Ahmed, Venkata Reddy Palleti, and Aditya P Mathur. Wadi: a water distribution testbed for research in the design of secure cyber physical systems. In *Proceedings of the 3rd international workshop on cyber-physical systems for smart water networks*, pages 25–28, 2017.

Juan Miguel Lopez Alcaraz and Nils Strodthoff. Diffusion-based time series imputation and forecasting with structured state space models. *Transactions on Machine Learning Research*, 2022.

Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A Zuluaga. Usad: Unsupervised anomaly detection on multivariate time series. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3395–3404, 2020.

Chris U Carmona, Francois-Xavier Aubet, Valentin Flunkert, and Jan Gasthaus. Neural contextual anomaly detection for time series. In *International Joint Conference on Artificial Intelligence*, 2022.

Wenchao Chen, Long Tian, Bo Chen, Liang Dai, Zhibin Duan, and Mingyuan Zhou. Deep variational graph convolutional recurrent network for multivariate time series anomaly detection. In *International Conference on Machine Learning*, pages 3621–3633, 2022.

Yuhang Chen, Chaoyun Zhang, Minghua Ma, Yudong Liu, Ruomeng Ding, Bowen Li, Shilin He, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. Imdiffusion: Imputed diffusion models for multivariate time series anomaly detection. *Proceedings of the VLDB Endowment*, 2024.

Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(9):10850–10869, 2023.

Enyan Dai and Jie Chen. Graph-augmented normalizing flows for anomaly detection of multiple time series. In *International Conference on Learning Representations*, 2022.

Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence*, pages 4027–4035, 2021.

Leyan Deng, Defu Lian, Zhenya Huang, and Enhong Chen. Graph convolutional adversarial networks for spatiotemporal anomaly detection. *IEEE Transactions on Neural Networks and Learning Systems*, 33(6):2416–2428, 2022.

Choubo Ding, Guansong Pang, and Chunhua Shen. Catching both gray and black swans: Open-set supervised anomaly detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7388–7398, 2022.

Xusheng Du, Jiong Yu, Zheng Chu, Lina Jin, and Jiaying Chen. Graph autoencoder-based unsupervised outlier detection. *Information Sciences*, 608:532–550, 2022.

Yuchen Fang, Jiandong Xie, Yan Zhao, Lu Chen, Yunjun Gao, and Kai Zheng. Temporal-frequency masked autoencoders for time series anomaly detection. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 1228–1241, 2024.

Alessandro Flaborea, Luca Collorone, Guido Maria D'Amely Di Melendugno, Stefano D'Arrigo, Bardh Prenkaj, and Fabio Galasso. Multimodal motion conditioned diffusion model for skeleton-based video anomaly detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10318–10329, 2023.

Jianfei Gao and Bruno Ribeiro. On the equivalence between temporal and static equivariant graph representations. In *International Conference on Machine Learning*, pages 7052–7076, 2022.

Karan Goel, Albert Gu, Chris Donahue, and Christopher Ré. It's raw! audio generation with state-space models. In *International Conference on Machine Learning*, pages 7616–7633, 2022.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in neural information processing systems*, 33:1474–1487, 2020.

Albert Gu, Goel, Karan, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*, 2022.

Antoine Guillot, Fabien Sauvet, Emmanuel H During, and Valentin Thorey. Dreem open datasets: Multi-scored sleep datasets to compare human and automated sleep staging. *IEEE transactions on neural systems and rehabilitation engineering*, 28(9):1955–1965, 2020.

Siho Han and Simon S Woo. Learning sparse latent graph representations for anomaly detection in multivariate time series. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 2977–2986, 2022.

Thi Kieu Khanh Ho and Narges Armanfard. Self-supervised learning for anomalous channel detection in eeg graphs: Application to seizure analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 7866–7874, 2023.

Thi Kieu Khanh Ho, Ali Karami, and Narges Armanfard. Graph anomaly detection in time series: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2025.

Hadi Hojjati, Mohammadreza Sadeghi, and Narges Armanfard. Multivariate time-series anomaly detection with temporal self-supervision and graphs: Application to vehicle failure prediction. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 242–259, 2023.

Kyle Hundman, Valentino Constantinou, Christopher Laporte, Ian Colwell, and Tom Soderstrom. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 387–395, 2018.

Xi Jiang, Jianlin Liu, Jinbao Wang, Qiang Nie, Kai Wu, Yong Liu, Chengjie Wang, and Feng Zheng. Softpatch: Unsupervised anomaly detection with noisy data. *Advances in Neural Information Processing Systems*, 35: 15433–15445, 2022.

Ali Karami, Thi Kieu Khanh Ho, and Narges Armanfard. Graph-jigsaw conditioned diffusion model for skeleton-based video anomaly detection. In *EEE/CVF Winter Conference on Applications of Computer Vision*, 2025.

Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. Towards a rigorous evaluation of time-series anomaly detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7194–7201, 2022.

Chung-Ming Kuan, Michalopoulos, Christos, and Zhijie Xiao. Quantile regression on quantile ranges–a threshold approach. *Journal of Time Series Analysis*, 38(1):99–119, 2017.

Thomas Lai, Thi Kieu Khanh Ho, and Narges Armanfard. Open-set multivariate time-series anomaly detection. In *European Conference on Artificial Intelligence*, 2024.

N Laptev, S Amizadeh, and Y Billawala. S5-a labeled anomaly detection dataset, version 1.0 (16m), 2015.

Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. Multivariate time series anomaly detection and interpretation using hierarchical inter-metric and temporal embedding. In *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining*, pages 3220–3230, 2021.

Haoran Liang, Lei Song, Junrong Du, Xuzhi Li, and Lili Guo. Consistent anomaly detection and localization of multivariate time series via cross-correlation graph-based encoder–decoder gan. *IEEE Transactions on Instrumentation and Measurement*, 71:1–10, 2021.

Feifei Liu, Chengyu Liu, Lina Zhao, Xiangyu Zhang, Xiaoling Wu, Xiaoyan Xu, Yulin Liu, Caiyun Ma, Shoushui Wei, Zhiqiang He, et al. An open access database for evaluating the algorithms of electrocardiogram rhythm and morphology abnormality detection. *Journal of Medical Imaging and Health Informatics*, 8(7):1368–1373, 2018.

Victor Livernoche, Vineet Jain, Yashar Hezaveh, and Siamak Ravanbakhsh. On diffusion modeling for anomaly detection. In *International Conference on Learning Representations*, 2024.

Ilya loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

Aditya P Mathur and Nils Ole Tippenhauer. Swat: A water treatment testbed for research and training on ics security. In *2016 international workshop on cyber-physical systems for smart water networks (CySWater)*, pages 31–36, 2016.

Kashif Rasul, Calvin Seward, Ingmar Schuster, and Roland Vollgraf. Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pages 8857–8868, 2021.

Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. Anomaly detection in time series: a comprehensive evaluation. *Proceedings of the VLDB Endowment*, 15(9): 1779–1797, 2022.

Vinit Shah, Eva Von Weltin, Silvia Lopez, James Riley McHugh, Lillian Veloso, Meysam Golmohammadi, Iyad Obeid, and Joseph Picone. The temple university hospital seizure detection corpus. *Frontiers in neuroinformatics*, 12:83, 2018.

Lifeng Shen, Zhuocong Li, and James Kwok. Timeseries anomaly detection using temporal hierarchical one-class network. *Advances in Neural Information Processing Systems*, 33:13016–13026, 2020.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265, 2015.

Nils Strodthoff, Patrick Wagner, Tobias Schaeffter, and Wojciech Samek. Deep learning for ecg analysis: Benchmarks and insights from ptb-xl. *IEEE Journal of Biomedical and Health Informatics*, 25(5):1519–1528, 2020.

Ya Su, Youjian Zhao, Chenhao Niu, Rong Liu, Wei Sun, and Dan Pei. Robust anomaly detection for multivariate time series through stochastic recurrent neural network. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2828–2837, 2019.

Yuting Sun, Guansong Pang, Guanhua Ye, Tong Chen, Xia Hu, and Hongzhi Yin. Unraveling the 'anomaly'in time series anomaly detection: a self-supervised tri-domain solution. In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, pages 981–994, 2024.

Siyi Tang, Jared Dunnmon, Khaled Kamal Saab, Xuan Zhang, Qianying Huang, Florian Dubost, Daniel Rubin, and Christopher Lee-Messer. Self-supervised graph neural networks for improved electroencephalographic seizure analysis. In *International Conference on Learning Representations*, 2021.

Siyi Tang, Jared A Dunnmon, Qu Liangqiong, Khaled K Saab, Tina Baykaner, Christopher Lee-Messer, and Daniel L Rubin. Modeling multivariate biosignals with graph neural networks and structured state space models. In *Conference on Health, Inference, and Learning*, pages 50–71, 2023.

Yusuke Tashiro, Jiaming Song, Yang Song, and Stefano Ermon. Csdi: Conditional score-based diffusion models for probabilistic time series imputation. *Advances in Neural Information Processing Systems*, 34:24804–24816, 2021.

Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena: A benchmark for efficient transformers. *arXiv preprint arXiv:2011.04006*, 2020.

Bowen Tian, Qinliang Su, and Jian Yin. Anomaly detection by leveraging incomplete anomalous knowledge with anomaly-aware bidirectional gans. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*, 2022.

Dennis Wagner, Tobias Michels, Florian C.F. Schulz, Arjun Nair, Maja Rudolph, and Marius Kloft. TimeseAD: Benchmarking deep multivariate time-series anomaly detection. *Transactions on Machine Learning Research*, 2023.

Yucheng Wang, Yuecong Xu, Jianfei Yang, Min Wu, Xiaoli Li, Lihua Xie, and Zhenghua Chen. Fully-connected spatial-temporal graph for multivariate time-series data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 15715–15724, 2024.

Yuanyuan Wei, Julian Jang-Jaccard, Wen Xu, Fariza Sabrina, Seyit Camtepe, and Mikael Boulic. Lstm-autoencoder-based anomaly detection for indoor air quality time-series data. *IEEE Sensors Journal*, 23(4):3787–3800, 2023.

Julia Wolleb, Florentin Bieder, Robin Sandkühler, and Philippe C Cattin. Diffusion models for medical anomaly detection. In *International Conference on Medical image computing and computer-assisted intervention*, pages 35–45, 2022.

Renjie Wu and Eamonn Keogh. Current time series anomaly detection benchmarks are flawed and are creating the illusion of progress. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How powerful are graph neural networks? In *International Conference on Learning Representations*, 2019.

Yiyuan Yang, Chaoli Zhang, Tian Zhou, Qingsong Wen, and Liang Sun. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3033–3045, 2023.

Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. Multivariate time-series anomaly detection via graph attention network. In *IEEE International Conference on Data Mining*, pages 841–850, 2020.

Yanqiao Zhu, Weizhi Xu, Jinghao Zhang, Yuanqi Du, Jieyu Zhang, Qiang Liu, Carl Yang, and Shu Wu. A survey on graph structure learning: Progress and opportunities. In *International Joint Conference on Artificial Intelligence*, 2022.

Contaminated Multivariate Time-Series Anomaly Detection with Spatio-Temporal Graph Conditional Diffusion Models (Appendix)

Thi Kieu Khanh Ho[1,2] **Narges Armanfard**[1,2]

[1]Department of Electrical and Computer Engineering, McGill University
[2]Mila - Quebec AI Institute, Montreal, QC, Canada

A RELATED WORK

A.1 TIME-SERIES ANOMALY DETECTION TECHNIQUES

Existing TSAD methods can be broadly categorized into supervised and unsupervised approaches. Early supervised methods require complete anomaly-labeled datasets that encompass the complete distribution of all potential anomalies during training Carmona et al. [2022], Tang et al. [2021, 2023]. However, acquiring labels for all possible anomalies is impractical for real-world applications. As a result, a recent work Lai et al. [2024] has explored TSAD in open-set scenarios Ding et al. [2022], Tian et al. [2022], where only a limited amount of labeled anomalous data is available during training. Although this reduces the dependency on complete anomaly-labeled data, open-set approaches still require a small number of labeled anomalies during training. Nonetheless, obtaining even a few labeled anomalies during data collection can be challenging in many cases. For example, epileptic seizures in brain recordings may occur unpredictably, often requiring prolonged monitoring to capture seizure events. Similarly, anomalies in industrial machinery, such as machine failures, are rare and might only occur after extended periods of operation.

To address these challenges, many studies have proposed unsupervised methods, assuming that only normal data is available during the training phase Su et al. [2019], Audibert et al. [2020], Shen et al. [2020], Deng and Hooi [2021], Li et al. [2021], Chen et al. [2022], Yang et al. [2023], Ho and Armanfard [2023], Hojjati et al. [2023], Chen et al. [2024]. Recent unsupervised methods, mostly leveraging deep learning architectures, can be further divided into four main categories based on the employed loss functions minimized during training: Autoencoder (AE)-based, Generative Adversarial Network (GAN)-based, Predictive-based, and Self-supervised methods Ho et al. [2025]. Specifically, AE-based methods utilize reconstruction loss to minimize the difference between the input data and its reconstructed output within an encoder-decoder framework Su et al. [2019], Chen et al. [2022]. GAN-based methods leverage the losses from both the generator and discriminator during training Liang et al. [2021], Deng et al. [2022]. Predictive-based methods utilize prediction error, aiming to forecast future values or properties rather than simply reconstructing the input data Zhao et al. [2020], Deng and Hooi [2021], Han and Woo [2022]. Lastly, self-supervised methods employ contrastive loss or other pretext-task-based losses to learn meaningful representations from the training data Ho and Armanfard [2023], Hojjati et al. [2023]. However, all these categories of unsupervised methods assume that the training data must be clean and do not address the challenges posed by contaminated data, limiting their effectiveness in accurately detecting anomalies in practical scenarios and applications.

A.2 DIFFUSION MODELS

Diffusion models, a class of generative models Sohl-Dickstein et al. [2015], Croitoru et al. [2023], have gained significant attention for anomaly detection across various domains. For example, Livernoche et al. [2024] introduced diffusion models to capture normal data distributions, enabling anomaly detection in tabular data. In image anomaly detection, Wolleb et al. [2022] proposed a diffusion model that demonstrates superior performance compared to the baseline methods, by generating normal image patterns, with anomalies detected as deviations from these patterns. In video anomaly detection, Flaborea et al. [2023] proposed a multimodal approach with diffusion models to detect anomalies in video data. Additionally, Karami et al.

[2025] proposed a diffusion model coupled with a graph-based module for modeling intra- and inter-variable dependencies existing in video data, to effectively detect anomalies.

In recent years, researchers have begun to explore the potential of diffusion models in the signal domain. For example, Tashiro et al. [2021] leveraged diffusion models for time-series imputation tasks, demonstrating strong performance in handling missing data. Likewise, Alcaraz and Strodthoff [2022], Rasul et al. [2021] applied diffusion models for both time-series imputation and forecasting tasks, showcasing their capability in achieving accurate future predictions. Notably, Chen et al. [2024] marked the first to adapt diffusion models specifically for TSAD tasks. Chen et al. [2024] considered diffusion models as time-series imputers to fill in missing data while simultaneously detecting anomalies. Despite these advancements, no existing studies have explored the use of diffusion models in the context of contaminated time-series data, where anomalies sneak into the normal training data. Addressing this challenge would unlock new potential for diffusion models, enabling them to tackle more complex real-world scenarios and applications.

A.3 LONG-TERM INTRA- AND INTER-VARIABLE DEPENDENCY MODELING

In time-series data, capturing both long-range intra- and inter-variable dependencies is essential for effective anomaly detectionHo et al. [2025]. Specifically, capturing long-range intra-variable dependencies is valuable for TSAD, as they help differentiate between normal and abnormal temporal variations. However, many existing methods overlook this critical aspect by preprocessing data into short-term intervals Carmona et al. [2022], Lai et al. [2024], Chen et al. [2024] or by relying on conventional temporal networks, such as recurrent neural networks Dai and Chen [2022], Hojjati et al. [2023] and transformers Yang et al. [2023], Chen et al. [2024]. Such data preprocessing steps may discard important information encoded in raw signals and neglect long-range intra-variable dependencies Tang et al. [2023]. Moreover, conventional temporal models struggle in handling sequences that are sampled at high sampling rates, leading to long sequences that can be up to thousands of time steps Tay et al. [2020]. Recently, advanced deep sequence models, such as structured state space models Gu et al. [2022], have demonstrated superior performance in long sequence modeling tasks, outperforming previous state-of-the-art models in speech classification Gu et al. [2022], audio generation Goel et al. [2022], and time-series forecasting Alcaraz and Strodthoff [2022]. Despite these successes, the potential of these models in unsupervised TSAD tasks remains unexplored. This presents an exciting opportunity for TSAD research to improve anomaly detection in complex and long-range time-series data.

Additionally, many recent studies have leveraged graphs to model inter-variable dependencies effectively. This property is valuable for TSAD as the changes of one variable can be used to predict the changes of another variable if they are related. For instance, several studies have proposed constructing static graphs based on prior knowledge, such as the Euclidean distances between sensors Tang et al. [2021, 2023], the correlation matrices of sensor features Ho and Armanfard [2023], or the mutual information between sensor features Hojjati et al. [2023] in multivariate sensory systems. However, this approach is often impractical for many real-world applications due to the dynamic nature of testing environments, e.g., in electroencephalogram monitoring of epilepsy patients, sensor placements may vary between subjects depending on many factors such as scalp injuries, epilepsy types and targeted brain regions. This results in variations in the inter-variable relationships across subjects, making it challenging for models employing static graphs to generalize effectively to diverse testing scenarios. Hence, dynamically learning the graphs is highly desirable as this approach allows for capturing the inter-variable dependencies over time Dai and Chen [2022], Deng and Hooi [2021], Han and Woo [2022], leading to improved anomaly detection performance in complex systems.

B THE PSEUDOCODES OF TSAD-C

To clearly outline the framework of TSAD-C and facilitate reproducibility, we present the pseudocode for the training phase, as well as the test phase of TSAD-C in Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 Training Phase of TSAD-C

Input: X_{train}, X_{valid}, $\mathbf{v}$, hyperparameters $\{\beta_0, \beta_T, T\}$, ξ_1, ξ_2, ξ_3.
Output: Learnable parameters Θ.

Compute $\mathbf{x}^u_{(i)} = \mathbf{x}_{(i)} \odot \mathbf{v}$, where $\mathbf{x}_{(i)} \in X_{\text{train}}$.
Compute diffusion hyperparameters $\bar{\alpha}_t$, $\bar{\beta}_t$, ϵ_t and c.
Compute $\mathbf{x}^t_{(i)}$.
Randomly initialize learnable parameters Θ.
while *not converged* **do**
  // Decontaminator
  Compute $\hat{\epsilon}_t = \epsilon_\theta(\mathbf{x}^t_{(i)}, t, c)$.
  $\mathcal{L}_{\text{noise}} = \|\epsilon_t \odot (1 - \mathbf{v}) - \hat{\epsilon}_t \odot (1 - \mathbf{v})\|^2$.
  Compute $\hat{\mathbf{x}}^0_{(i)}$.
  // Long-range Variable Dependency Modeling
  Compute $\mathbf{H}_{(i)} = \text{S4}(\hat{\mathbf{x}}^0_{(i)})$.
  Construct $\mathbb{G}_{(i)} = \{\mathcal{G}^m_{(i)}\}_{m=1}^d$.
  $\mathcal{L}_{\text{graph}} = 0$.
  for $m \in 1, 2, ..., d$ **do**
    Construct $\mathcal{G}^m_{(i)} = \{\mathbf{E}^m_{(i)}, \mathcal{A}^m_{(i)}\}$, where $\mathbf{E}^m_{(i)} = \text{mean-pool}(\mathbf{H}_{(i)})$, $\mathcal{A}^m_{(i)} = \text{Graph-Learning}(\mathbf{E}^m_{(i)}, \mathcal{A'}^m_{(i)})$.
    $\mathcal{L}_{\text{graph}} = \mathcal{L}_{\text{graph}} + \xi_1 \mathcal{L}_{\text{smooth}}(\mathbf{E}^m_{(i)}, \mathcal{A}^m_{(i)}) + \xi_2 \mathcal{L}_{\text{sparse}}(\mathcal{A}^m_{(i)}) + \xi_3 \mathcal{L}_{\text{connect}}(\mathcal{A}^m_{(i)})$.
    Compute $\mathbf{z}^m_{(i)} = \text{GIN}(\mathbf{E}^m_{(i)}, \mathcal{A}^m_{(i)})$.
  end for
  Compute $\mathbf{Z}_{(i)} = \text{concat}(\mathbf{z}^1_{(i)}, \ldots, \mathbf{z}^d_{(i)})$.
  Compute $\hat{\hat{\mathbf{x}}}_{(i)} = \text{Linear}(\mathbf{Z}_{(i)})$.
  $\mathcal{L}_{\text{recon}} = \|\hat{\mathbf{x}}^0_{(i)} - \hat{\hat{\mathbf{x}}}_{(i)}\|^2$.
  $\mathcal{L} = \mathcal{L}_{\text{noise}} + \mathcal{L}_{\text{graph}} + \mathcal{L}_{\text{recon}}$.
  Backpropagate $\mathcal{L}$ to update Θ.
  Early stopping using X_{Valid}.
end while

Algorithm 2 Inference Phase of TSAD-C

Input: X_{test}, X_{valid}, $\mathbf{v}$, hyperparameters $\{\beta_0, \beta_T, T\}$, λ_1, λ_2.
Output: Anomaly score s.

// Decontaminator
Compute $\mathbf{x}^u_{(i)} = \mathbf{x}_{(i)} \odot \mathbf{v}$, where $\mathbf{x}_{(i)} \in X_{\text{test}}$.
Compute diffusion hyperparameters $\bar{\alpha}_t$, $\bar{\beta}_t$, ϵ_t and c.
Compute $\mathbf{x}^T_{(i)}$.
for $t = T, T-1, \ldots, 1$ **do**
  Compute $\mu_\theta(\mathbf{x}^t_{(i)}, t, c)$ and $\sigma_\theta(\mathbf{x}^t_{(i)}, t, c)$.
  Sample $\mathbf{x}^{t-1}_{(i)} \sim q_\theta(\mathbf{x}^{t-1}_{(i)} | \mathbf{x}^t_{(i)}) \coloneqq \mathcal{N}(\mathbf{x}^{t-1}_{(i)}; \mu_\theta(\mathbf{x}^t_{(i)}, t, c), \sigma_\theta(\mathbf{x}^t_{(i)}, t, c)^2\mathbf{I})$.
end for
return $\hat{\mathbf{x}}^0_{(i)}$.
// Long-range Variable Dependency Modeling
Compute $\mathbf{H}_{(i)}$, $\mathbb{G}_{(i)}$, $\mathbf{Z}_{(i)}$.
Compute $\hat{\hat{\mathbf{x}}}_{(i)}$.
// Anomaly Scoring
Compute $s_1 = \text{RMSE}(\hat{\mathbf{x}}^0_{(i)}, \mathbf{x}_{(i)})$.
Compute $s_2 = \text{RMSE}(\hat{\hat{\mathbf{x}}}_{(i)}, \mathbf{x}_{(i)})$.
Compute $s = \lambda_1 s_1 + \lambda_2 s_2$.
Search τ based on the unlabeled X_{valid}.
if $s > \tau$ **then**
  $\mathbf{x}_{(i)}$ is an anomaly.
else
  $\mathbf{x}_{(i)}$ is a normal observation.
end if

C DATASETS

In this section, we describe four reliable datasets collected from various domains that are used in our study, including SMD, ICBEB, DODH and TUSZ datasets.

C.1 SMD

The Server Machine Dataset (SMD) Su et al. [2019] is a multivariate time-series dataset comprising five weeks of data from 28 different server machines collected by a large Internet company. SMD is widely used in the TSAD field and is considered of much higher quality Wagner et al. [2023] compared to other benchmarks such as Yahoo Laptev et al. [2015], NASA Hundman et al. [2018], SWaT Mathur and Tippenhauer [2016], WADI Ahmed et al. [2017], SMAP Hundman et al. [2018], and MSL Hundman et al. [2018]. However, several concerns have been raised about this dataset. First, Wu and Keogh [2021] highlighted the triviality issue in SMD, noting that the dataset can be easily solved by just one line of code. Second, according to Wagner et al. [2023], SMD exhibits a distributional shift issue, where there is a shift in the statistical distribution between the training and test sets. The authors identified and excluded several machines from the dataset upon observing significant changes in mean and standard deviation between the training and test sets for those machines. They then asserted that their modified dataset is more suitable for benchmarking TSAD algorithms.

Aware of these issues, we implement data preprocessing steps that differ significantly from previous studies Su et al. [2019], Li et al. [2021], Chen et al. [2022], Yang et al. [2023], Chen et al. [2024] to address those concerns. First, we divide the dataset into observations (intervals), each with a length of 600. We then perform interval-level anomaly detection, as defined in our problem definition in **Section 2** - Paragraph 1 of the main paper, aligning with recent standards in TSAD Carmona et al. [2022], Lai et al. [2024]. This approach contrasts with point-level anomaly detection, which introduces point-adjustment bias – a common issue in the TSAD field Kim et al. [2022], i.e., without the point-adjustment strategy, the performance of existing methods cannot even outperforms a random baseline.

Second, we implicitly and partially mitigate the triviality issue by treating the time series as a multivariate series, emphasizing the importance of capturing inter-variable dependencies, rather than focusing solely on individual sensors as commonly

done in previous studies Su et al. [2019], Yang et al. [2023]. To build a robust and generalized method, we also combine all 28 machines and train only a single model for both our proposed method and compared methods. Note that, previous studies Su et al. [2019] suggested that the data from each of the 28 machines should be trained and tested separately. Additionally, by adopting interval-level anomaly detection instead of point-level anomaly detection, we increase the complexity and challenge for TSAD algorithms. Lastly, to address the distributional shift issue, we follow the procedure outlined in Wagner et al. [2023], and examine the mean and standard deviation of the training and test sets to minimize any shift in the dataset.

Given these crucial procedures, we ensure a more rigorous and robust evaluation of TSAD algorithms on SMD. Hence, the results of our implementation on compared methods may differ from those originally reported in their papers.

C.2 ICBEB

The International Conference on Biomedical Engineering and Biotechnology (ICBEB) Liu et al. [2018] is an electrocardiogram (ECG) dataset. It consists of normal data and different types of abnormal cardiac disorders. Each recording is annotated by up to three ECG experts and might be associated with more than one abnormal classes. We select the normal ECG waveforms as normal data, while five abnormal types, including atrial fibrillation, first-degree atrioventricular, right bundle branch, premature ventricular contraction and ST-segment elevation, are selected as abnormal data. We follow the preprocessing steps as described in Strodthoff et al. [2020], Tang et al. [2023]. The sampling frequency rate is 100Hz, resulting in observations with the length of 6,000 time steps. Each observation consists of 12 sensors. Our task is to detect abnormal cardiac observations from all five types that deviate from the normal cardiac activities.

C.3 DODH

The Dreem Open Dataset Healthy, called DODH Guillot et al. [2020] is a sleep dataset collected from polysomnographic (PSG) signals of 25 volunteers. Each recording includes different sleep stages, namely Awake, rapid eye movement (REM), and non-REM sleep stage N3, and is annotated by a consensus of five experienced PSG readers. We select N3 (the deepest sleeping stage) as normal data since the brain activity during this stage has an identifiable pattern of delta waves; and body activities such as breathing and muscle tone rate decrease. Meanwhile, REM and Awake stages are abnormal since the brain activity during these periods rises up; and the body activities such as the eyes and the muscles start increasing. The PSG signals are recorded from different locations across the body (brain-eyes-heart-legs), presenting a very challenging problem for TSAD algorithms due to inconsistent signal patterns. There is a total of 16 recording sensors: 12 EEG sensors, 1 electromyographic (EMG) sensor, 2 electrooculography (EOG) sensors, and 1 ECG sensor. Following the preprocessing steps described in Tang et al. [2023], we resample PSG signals with a sampling frequency rate of 250Hz, leading to a length of 7,500 time steps for every observation. Our task is to detect abnormal sleep observations from two types that deviate from the normal sleep activities.

C.4 TUSZ

The University Hospital Seizure Detection Corpus, called TUSZ Shah et al. [2018] is the largest public electroencephalogram (EEG) seizure dataset to date. It contains 5612 EEG files with 3050 annotated seizures, and different types of seizures are defined although some of them are less presented. Annotations are made through a consensus of at least three experienced EEG readers. For each patient, there are several sessions that consist of files related to one or more recordings. For each recording, there are five EDF files containing the raw EEG signals and a header indicating frequency, duration and date. In our experiments, we select resting states of the brain activities as normal data, while abnormal data is collected from two types of seizures (focal and generalized). Each EEG recording consists of 19 sensors. Note that compared to other types of signals such as ECG that represents fairly predictable heart waveforms, TUSZ contains much more stochastic signals due to various physiological states in the brain. Following prior studies on seizure analysis Tang et al. [2021], Ho and Armanfard [2023], we resample all EEG signals with a sampling frequency rate of 200Hz, resulting in a length of 12,000 time steps for each observation. Our task is to detect whether there exist abnormal patterns from two types of seizures in the observation.

It is important to re-emphasize that **ICBEB**, **DODH** and **TUSZ** are well-established yet challenging datasets and have not received criticisms in the TSAD field. To minimize human labeling errors, they are annotated by a panel of 3-5 experts, who have practical experience dealing with patients in clinical settings, and take into account factors such as patient history, symptoms, diagnostic tests, and treatment outcomes when identifying anomalies. Additionally, they often exhibit long-range intra-variable dependencies, reflecting phenomena like physiological states or disease progression, which oversimplified

models cannot capture complex time-series dynamics. Importantly, unlike existing datasets such as **SMD** that contain a single anomaly type, these three datasets reflect real-world scenarios with diverse anomaly types, where each type may have distinct characteristics or exhibit patterns similar to normal variability.

Note that for all datasets, we first split the data at the machine/patient level into three subsets: X_{train}, X_{valid}, and X_{test}. This means each machine/patient's data is entirely contained within one of these subsets to ensure independence between them. Within each machine/patient, each observation is collected using a non-overlapping split to avoid data leakage.

D BASELINES

As mentioned in the main paper, we compare TSAD-C with 12 SOTA unsupervised methods in the TSAD field. These methods can be categorized into three groups as follows:

- Methods capturing Intra-variable dependencies include:
  - **USAD** Su et al. [2019] is a Generative Adversarial Network consisting of two Autoencoders. It is trained in a two-phase process: the first phase reconstructs the normal data, while the second phase distinguishes the real data from the data coming from the first Autoencoder. The anomaly score is a combination of the scores obtained from both phases.
  - **LSTM-AE** Wei et al. [2023] is a Long Short-Term Memory based Autoencoder. The anomaly score is the reconstruction error.
  - **S4-AE** Gu et al. [2022] is a Structured State Space based Autoencoder. The anomaly score is the reconstruction error.
  - **DCdetector** Yang et al. [2023] is a Transformer based framework consisting of two self-attention networks. The first is for a wider patch-wise view and the another is for a finer in-patch view. The Kullback-Liebler contrastive loss minimizes the distance between the two views for normal data, with the assumption that the views will be different for abnormal data. The anomaly score is computed based on the contrastive loss.
- Methods capturing Inter-variable dependencies include:
  - **GAE** Du et al. [2022] is a Graph-based Autoencoder that learns the graph structure and features during training. The reconstruction error is used as the anomaly score.
  - **GDN** Deng and Hooi [2021] is a Predictive-based approach that employs two modules: a graph structure learning module to capture the sensor relationships and a graph attention-based forecasting module to predict future values of every sensor. The anomaly score is computed by the prediction error via a graph deviation scoring module.
  - **EEG-CGS** Ho and Armanfard [2023] is a Self-supervised approach that first constructs graphs based on the correlations between sensors. It then leverages local structural and contextual information within the graphs to generate positive and negative sub-graphs. EEG-CGS is trained by minimizing contrastive and generative losses, with the anomaly score derived from a combination of these losses.
- Methods capturing Both-variable dependencies include:
  - **InterFusion** Li et al. [2021] is a Variational Autoencoder based approach with two stochastic latent variables, each of which learns low-dimensional inter- or intra-variable embeddings within the normal data. The reconstruction error is considered as the anomaly score.
  - **DVGCRN** Chen et al. [2022] is a Variational Autoencoder based approach designed to capture multilevel intra- and inter-variable information within the raw data, and the inter-variable information in the latent space. It comprises two main components to model these relationships: a forecasting model for single-time step prediction, and a reconstruction model for reconstructing the input data. The anomaly score is a combination of prediction and reconstruction errors.
  - **GRU-GNN** Tang et al. [2023] incorporates a Gate Recurrent Unit network and a Graph Neural Network model to extract both intra-variable inter-variable dependencies in time-series data. The reconstruction score is used as the anomaly score for each observation.
  - **GraphS4mer** Tang et al. [2023] is a Transformer based approach that incorporates S4 layers to extract long-range intra-variable dependencies, a graph structure learning technique based on an attention mechanism to construct graphs and Graph Neural Network layers to capture the inter-variable dependencies. The reconstruction score is used as the anomaly score.

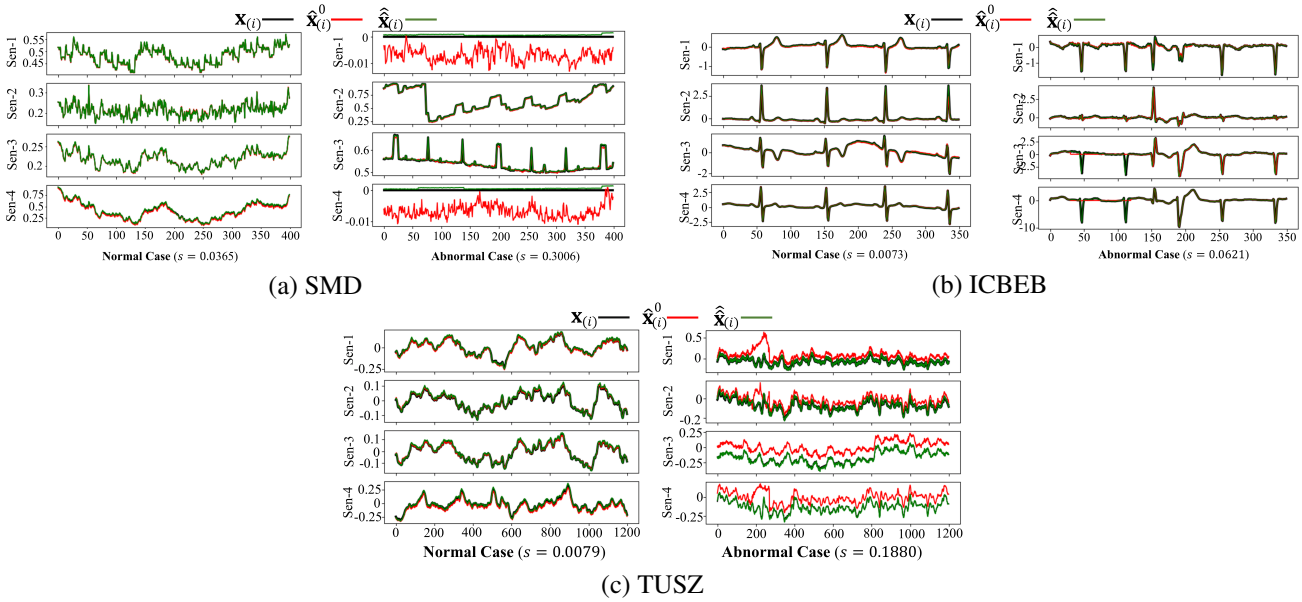

Figure 5: Comparison between normal and abnormal cases for the masked segment in (a) SMD and (b) ICBEB and (c) TUSZ. The masked strategy used is BoM. Each case includes the ground truth $\mathbf{x}_{(i)}$, the decontaminated data $\hat{\mathbf{x}}^0_{(i)}$ and the reconstructed data $\hat{\hat{\mathbf{x}}}_{(i)}$.

- **IMDiffusion** Chen et al. [2024] proposes a Diffusion model as a time-series imputer to capture intra- and inter-variable dependencies. It employs a Transformer network for noise estimation. In the test phase, ImDiffusion utilizes the prediction error at each denoising step and ensembles them using a voting function to determine the final anomaly scores.

Note that none of the existing methods incorporate a mechanism to handle contaminated data, i.e., real-world anomalies contaminate normal data during the training phase, as they all assume that the training data must be clean. We implement these baselines with their default model architecture, optimization procedure, and hyperparameters as suggested by the authors of the original papers. To ensure a fair comparison, we use the same datasets, experimental settings and evaluation protocols for TSAD-C and all compared methods.

E IMPLEMENTATION DETAILS

We set the masking ratio to match the anomaly ratio η in X_{train}. Thus, for each dataset, r is determined as the product of the observation length L and η as detailed in Table 1. We set Γ to L for each dataset, $d = 6$, $g = \frac{\Gamma}{d}$, $U = 128$, $\delta = 3$, $\zeta = 0.6$, $\xi_1 = 1$, $\xi_2 = 0.05$, and $\xi_3 = 0.5$. Additionally, we set $\beta_0 = 10^{-4}$, $\beta_T = 0.02$, $T = 50$, $\lambda_1 = 0.01$, and $\lambda_2 = 1.2$. The dimension of S4 states is fixed at 64. The batch size remains consistent at 4 for all datasets. The entire TSAD-C network is optimized via the AdamW optimizer Loshchilov and Hutter [2019], with a cosine learning rate scheduler loshchilov and Hutter [2017], initialized with a learning rate of 8e-4. Model training would be early stopped if the validation loss fails to decrease for 20 consecutive epochs, with a maximum of 100 epochs. It is worth noting that to ensure a fair comparison, these hyperparameters are fixed across all experiments and datasets. All experiments are conducted on a single NVIDIA V-100 GPU (32 GB).

F ADDITIONAL VISUALIZATIONS

Figures 5 (a), (b) and (c) show visualizations that compare the ground truth $\mathbf{x}_{(i)}$, decontaminated data $\hat{\mathbf{x}}^0_{(i)}$, and reconstructed data $\hat{\hat{\mathbf{x}}}_{(i)}$ in SMD, ICBEB and TUSZ, respectively. It is shown that $\hat{\mathbf{x}}^0_{(i)}$ and $\hat{\hat{\mathbf{x}}}_{(i)}$ fit $\mathbf{x}_{(i)}$ very well in the normal case, leading to a lower s. Meanwhile, there are significant fluctuations of $\hat{\mathbf{x}}^0_{(i)}$ and $\hat{\hat{\mathbf{x}}}_{(i)}$ compared to $\mathbf{x}_{(i)}$ in the abnormal case, resulting in a much higher s. These observations are consistent across all datasets. This shows the effectiveness of TSAD-C

in distinguishing anomalies from normal data.

G COMPUTATIONAL COST AND SCALABILITY ANALYSIS

Due to the diversity in dataset characteristics and dimensions described in **Appendix C** and Table 1, in this section, we present dedicated discussions on TSAD-C's computational cost and scalability. Following the TSAD literature Chen et al. [2022], Li et al. [2021], Wang et al. [2024], we report the Floating-point Operations (FLOPs), the training time per epoch, the testing time per sample, and the detection performance of TSAD-C in Table 6. Additionally, we conduct a detailed efficiency comparison between TSAD-C and comparable methods, i.e., InterFusion and IMDiffusion, which similarly handle both intra- and inter-variable dependencies, as discussed in **Appendix D**. It is shown that TSAD-C requires the fewest FLOPs. Notably, the training and testing times of TSAD-C are much faster than those of compared methods, while achieving significant improvements in detection performance. TSAD-C also maintains manageable training and testing durations, scaling reasonably as the dataset grows, when utilizing a single GPU. This efficiency is achieved despite significant dataset variations due to several reasons. First, our Decontaminator minimizes the noise error on only masked portions to have a simpler and more streamlined training process. The decontaminated data is then obtained in a *single* step during the reverse process in the training phase, speeding up data preparation for subsequent modules – a significant advantage for practical applications. Detailed studies are shown in Section 3.2.5.

Additionally, as mentioned in Section 2.2, since each observation can encompass thousands of timestamps (e.g., ICBEB with 6,000 timestamps, DODH with 7,500 timestamps and TUSZ with 12,000 timestamps), constructing a graph for every time step becomes inefficient and computationally demanding. To address this, we construct a graph over a defined time window, determined by the hyperparameter g, which aids in information aggregation. This strategy not only leads to a graph with reduced noise but also facilitates faster computations. Moreover, our graph module focuses on local connectivity patterns by δ-nearest neighbors, which allow to scale efficiently because each node's computation is limited to its neighbors, reducing the overall computational complexity compared to fully-connected networks Wang et al. [2024]. We also include the regularization terms in $\mathcal{L}_{\text{graph}}$ to regularize the graphs, as shown in Equation (8). For instance, including $\mathcal{L}_{\text{sparse}}$ helps avoid overly connected graphs, which reduces computational costs for very large datasets like TUSZ. This regularization also aids in maintaining computational efficiency while preserving the graph's ability to capture relevant data patterns. These aspects ensure TSAD-C's efficiency to varying dataset sizes without compromising on detection performance.

Table 6: Efficiency comparison between existing methods and TSAD-C in terms of FLOPs, the training time per epoch, the testing time per sample, and the detection performance (min: minutes, sec: seconds). The best scores are highlighted in bold.

Method	**Dataset**	FLOPs	Training time per epoch (min)	Testing time per sample (sec)	Performance	
					F1	APR
InterFusion	SMD	4.6G	4.2	35	0.383	0.490
	ICBEB	11.8G	19.1	73	0.649	0.753
	DODH	20.4G	29.4	91	0.418	0.559
	TUSZ	46.2G	48.4	126	0.532	0.628
IMDiffusion	SMD	5.8G	5.4	69	0.426	0.533
	ICBEB	20.4G	20.7	94	0.611	0.750
	DODH	32.2G	31.5	116	0.544	0.651
	TUSZ	69.7G	47.6	157	0.381	0.532
TSAD-C	SMD	2.3G	1.8	17	**0.479**	**0.604**
	ICBEB	7.3G	12.2	32	**0.707**	**0.773**
	DODH	12.2G	19.3	66	**0.652**	**0.728**
	TUSZ	32.1G	38.4	84	**0.545**	**0.652**