Beyond Marginals: Learning Joint Spatio-Temporal Patterns for Multivariate Anomaly Detection

Anonymous authors
Paper under double-blind review

Abstract

In this paper, we aim to improve anomaly detection (AD) by incorporating the *time-varying non-linear spatio-temporal correlations* of the multi-variate time series data in the modeling process. In multivariate AD, the *simultaneous deviation* of multiple nodes from their expected behavior can indicate an anomaly, even if no *individual* node shows a clearly abnormal pattern. In many existing approaches, time series variables are assumed to be (conditionally) independent, which oversimplifies real-world interactions. Our approach addresses this by modeling joint dependencies using a copula-based framework, which decouples the modeling of *marginal distributions, temporal dynamics, and inter-variable dependencies.* We use a transformer encoder to capture temporal patterns, and to model spatial (inter-variable) dependencies, we integrate a copula. Both components are trained jointly in a latent space using a self-supervised contrastive learning objective to learn meaningful feature representations to separate normal and anomaly samples.

1 Introduction

Modern industrial systems rely on networks of interconnected sensors that produce vast streams of multivariate time series data during operation. Detecting anomalies in these data plays a critical role in identifying faults early, mitigating security threats, and maintaining system reliability and safety. In industrial settings, time-series data are frequently used to monitor the performance of machines, IT infrastructure, spacecraft, and engines. Anomaly detection has become a vital component of time series analysis, enabling early detection of faults and preventing potential failures. Recent advances in deep learning have spurred the development of various methods to address this problem. For instance, recurrent neural networks (RNNs) Hundman et al. (2018); Su et al. (2019); Canizo et al. (2019) have been widely used to capture temporal dependencies in multivariate sequences. Meanwhile, other approaches employ graph-based models or Transformer architectures Vaswani (2017) to focus on variable relationships and sequential patterns Deng & Hooi (2021), Anomaly Transformer Xu (2021). These models effectively utilize temporal structures and adapt neural networks to time series tasks. Despite these advancements, the detection of anomalies in multivariate time series data remains a challenge. The primary difficulty arises from the intricate temporal dependencies and correlations between multiple variables. Anomalies often manifest as subtle deviations that are hard to isolate from natural fluctuations without contextual awareness. In addition, real-world datasets frequently suffer from noise, missing values, and high dimensionality, further complicating the modeling process. A major limitation in this domain is the scarcity of labeled data. In many cases, it is unclear during training whether a given point represents an anomaly. This lack of ground-truth labels has driven the adoption of unsupervised learning approaches. Methods such as autoencoders and adversarial networks attempt to model data distributions without labels to identify deviations. However, unsupervised approaches often struggle with contextual anomalies and dependencies between variables, making detection unreliable in complex scenarios.

Multivariate time-series data, especially high-order multivariate time series (HO-MTS), introduce additional layers of complexity that make anomaly detection particularly challenging. Unlike univariate time series, which involve a single variable observed over time, HO-MTS captures interdependencies between multiple variables, not just at a single time step but also across multiple time lags. This temporal and cross-variable dependency structure amplifies the difficulty of modeling and detecting anomalies. HO-MTS data require models to account for both spatial correlations (relationships between variables) and temporal dependencies (relationships across time steps). For instance, a sensor measuring pressure at time t might depend on the temperature reading at time $t-1$ or flow rate at time $t-2$. These dependencies often span long time horizons, making it necessary to handle lagged interactions effectively. Traditional models, such as autoregressive methods, struggle to capture such intricate relationships, particularly when the data have nonlinear patterns or dependencies that are not explicitly observable. This challenge is compounded when anomalies arise from unexpected combinations of variable interactions rather than simple threshold violations, requiring models to analyze contextual anomalies instead of point anomalies. In HO-MTS, anomalies can occur as contextual deviations rather than isolated outliers. For example, a sudden spike in temperature may not be anomalous if it follows an increase in pressure, but it could indicate a fault if the pressure remains constant. Anomalies may involve correlated changes between multiple variables rather than deviations in a single variable, making it harder to detect them without modeling joint distributions.

Supervised learning methods, on the other hand, present an attractive alternative when labeled data is available. These approaches can explicitly learn patterns associated with anomalies and differentiate them from normal behavior, especially when anomalies are subtle or involve relationships between multiple variables. Industrial datasets often contain very few labeled anomalies. Recent techniques, such as semi-supervised learning and contrastive learning, enable models to utilize a small set of labeled data while benefiting from larger unlabeled datasets. Supervised techniques also benefit from the ability to incorporate domain knowledge through labeled examples, improving interpretability and performance. Pre-trained models and data augmentation techniques can leverage small amounts of labeled data to improve performance significantly. Although unsupervised methods are useful when labeled data is unavailable, supervised learning provides better performance in scenarios where labeled data can be obtained (even in small quantities). It can model complex dependencies, improve interpretability, and leverage domain-specific insights. Detecting anomalies in HO-MTS data requires capturing both temporal dependencies and spatial (variable) correlations effectively. While deep learning models such as Long Short-Term Memory networks (LSTMs) and Transformers have shown success in modeling temporal patterns, they often struggle to adequately capture latent dependencies across variables, especially in scenarios with complex interactions and nonlinear relationships.

Transformers Vaswani (2017) rely on self-attention mechanisms to capture long-range dependencies in sequences. While attention allows modeling dependencies across time and variables, it does so explicitly at the input level and may fail to implicitly learn latent structures among variables, especially in high-dimensional data where dependencies are often hierarchical or nonlinear. Recent studies Deng & Hooi (2021); Xu et al. (2023) demonstrate that Transformers can improve modeling variable interactions compared to RNNs, but they still struggle when anomalies stem from latent feature correlations that are not easily captured through direct feature interactions. The above limitations motivate the need for latent space dependency modeling techniques that go beyond surface-level attention or sequential memory. Anomalies in multivariate time series often emerge from hidden dependencies among variables in latent representations, rather than observable patterns in raw features.

To address these challenges, copula-based methods offer a promising approach to model latent dependencies explicitly. Copulas separate the marginal distributions of variables from their dependency structure Salinas et al. (2019). This is particularly useful in multivariate settings where anomalies may arise due to joint dependencies rather than individual deviations. This enables models to preserve both spatial (cross-variable) and temporal dependencies simultaneously in a lower-dimensional representation. Our contribution can be summarized as follows:

- We introduce an end-to-end training framework that jointly optimizes a Transformer encoder for extracting temporal patterns in high-dimensional time series and the copula parameters for capturing dependencies in the resulting latent space. By backpropagating through both components, the model discovers latent embeddings that preserve local and long-range time relations while also conforming to consistent variable-to-variable dependencies.
- Instead of labeling individual samples, we treat each window or frame as a coherent sequence, then build a contrastive loss that enforces normal frames to achieve a high copula log-likelihood,

while anomalous frames are pushed below a margin in log-likelihood space. This approach better captures short-range temporal structure and ensures that anomalies which exhibit subtle multivariate deviations are effectively separated in the latent embedding space.

- Experiments conducted on multiple public benchmark multivariate time series datasets demonstrate that our model consistently outperforms state-of-the-art techniques, achieving higher precision, recall, and AUC-ROC scores of 5- 18%.

1.1 Related Work

Time series anomaly detection has been approached using both supervised Jia et al. (2019),Cook et al. (2019) and unsupervised methods Audibert et al. (2020), Zhang et al. (2021), Thill et al. (2021), each catering to different challenges posed by the data. Unsupervised methods dominate this field due to the lack of labeled anomalies in most real-world datasets. Techniques such as autoencoders, isolation forests, and Gaussian mixtures focus on modeling the distribution of normal data and identifying deviations as anomalies. Deep learning models, such as LSTMs and Autoencoders, leverage reconstruction errors or forecasting residuals to detect anomalies without requiring labels. However, these methods often struggle with contextual and correlated anomalies, especially in multivariate settings where relationships between variables evolve over time. Supervised methods, on the other hand, utilize labeled datasets to explicitly distinguish anomalies from normal patterns. Approaches such as RNN classifiers, attention-based networks Wang & Liu (2024); Zhao et al. (2020), and graph neural networks Zhao et al. (2020), Deng & Hooi (2021) excel at learning complex dependencies and classifying anomalies when labeled data is available. Recent advances in semi-supervised learning Akcay et al. (2019) and transfer learning have further extended supervised approaches to scenarios with limited labeled data, offering improved accuracy and interoperability. While unsupervised methods are widely used due to their flexibility, supervised approaches are gaining traction as they can better capture latent dependencies and nonlinear correlations in high-order multivariate time series, particularly when integrated with copula-based models to enhance dependency modeling in latent spaces. Previously, encoder-decoder based architectures integrated with adversarial training framework Audibert et al. (2020) that leveraged the strengths of both autoencoders and adversarial training while addressing the shortcomings of each approach were developed for multivariate time series anomaly detection.

TranAD Tuli et al. (2022) leverages attention-based encoders, self-conditioning, adversarial training, and MAML for robust, efficient, and data-efficient anomaly detection and diagnosis. STADN Tian et al. (2023) integrates spatial-temporal information using graph attention and LSTM networks, predicts sensor behavior, and enhances anomaly detection by reconstructing prediction errors for better discrimination. Anomaly-BERTJeong et al. (2023) addresses this issue by introducing a data degradation scheme for self-supervised model training. They specifically define four types of synthetic outliers and propose a degradation process in which parts of the input data are replaced with these outliers. In addition to leveraging the self-attention mechanism of the Transformer architecture, their approach transforms multivariate data points into temporal representations enriched with relative position bias and computes anomaly scores based on these representations. TACTis Drouin et al. (2022) addresses the challenge of estimating the joint predictive distribution for high-dimensional multivariate time series by introducing a flexible approach built on the transformer architecture, leveraging an attention-based decoder that is theoretically proven to replicate the behavior of non-parametric copulas. Anomaly-Transformer Xu (2021) identifies anomalies by leveraging their tendency to form concentrated associations with adjacent points, unlike normal data which associates more broadly across the series. CARLA Darban et al. (2025) leverages contrastive learning and a self-supervised strategy to enhance anomaly detection by learning similar representations for adjacent windows and distinguishing anomalies based on proximity to the representation. Another recent work Tang et al. (2024) addressed the problem of time series anomaly detection with self-supervised contrastive learning by designing a perturbation classifier to infer the pseudo-labels of data perturbations.

2 Proposed Framework

In this section, we elaborate the deep learning framework, contrastive loss function and the training strategy that we considered to develop our model.

2.1 Problem Statement

In our problem, we consider a multivariate time series $\mathbf{X} \in \mathbb{R}^{D\times T}$ arising from a cyber-physical system, where D is the dimension of each time frame and T is the total length of the time series. We embed $\mathbf{X}$ into an embedding space $\mathbf{Z} \in \mathbb{R}^{d\times T}$, capturing the most influential features. Our goal is to design a contrastive loss that leverages the *joint distribution* (or *mutual information*) among these latent variables, as measured via a *copula log-density*, to effectively *separate* normal from anomalous frames. Intuitively, normal data lies in a *higher log-likelihood of the copula density function* region in the latent space, reflecting consistent dependency patterns, whereas anomalies disrupt these dependencies and thus spread over a *lower log-likelihood of the copula density function*. By constructing a contrastive objective that enforces a large gap in copula log-density (or mutual information) between normal and anomalous frames, we exploit the fundamental distinction in their dependence structures to enhance *anomaly detection* within the latent feature space.

In this approach, each short subsequence of length L in the multivariate time series is labeled as a *frame*, which can be either normal (label 0) or anomalous (label 1). *(A) Labeling Frames.* If any timestamp in a length-L snippet is anomalous, then the entire frame is labeled as anomaly; otherwise, it is normal. This ensures that anomalies spanning multiple time steps are not fragmented or overlooked.

(B) Generating Frame Pairs. Once frames are labeled, we form pairs $(\mathbf{x}_i, \mathbf{x}_j)$ that may be normal–normal, normal–anomaly, or anomaly–anomaly. Such pairs can then be used in contrastive or mutual-information-based learning, since each pair directly encodes local temporal dependencies and interactions among variables. If either sequence is anomalous, we label the pair as anomaly-involved.

(C) Advantages for Time-Series. By treating small windows as frames rather than single samples, we can better capture short-range temporal correlations, especially when anomalies or key dependency patterns span multiple points.

(D) Estimating Copula log-density. Computing mutual information or fitting a copula requires a sufficient number of samples and a joint view of multiple dimensions; having short subsequences provides richer data for these estimations. This approach results in more stable estimates of dependency patterns, allowing the model to differentiate high-log-density regions (representing normal data) from low-log-density regions (representing anomalies).

2.2 Transformer Encoder

In this setting, we employ a Transformer encoder to project: $f_\theta : \mathbb{R}^{D\times T} \to \mathbb{R}^{d\times T}, \quad \mathbf{X} \mapsto \mathbf{Z}$. A Transformer processes the entire time series in parallel by applying self-attention across all time steps. This mechanism allows every position in the sequence to reach every other position, capturing both local and long-range dependencies. In particular, for longer time horizons, the model is less prone to forgetting distant signals compared to recurrent networks. Hence, each latent vector $\mathbf{z}_t$ (corresponding to a snippet or frame in time) encodes the key temporal patterns that matter to distinguish normal from anomaly sequences. Each dimension of $\mathbf{z}$ emerges from attention-based feature extraction, so normal data cluster around consistent patterns, while anomalies, which break typical variable interactions, are projected elsewhere in the latent space. This dimensionality reduction clarifies which features (and which timesteps) are most crucial to anomaly detection. We use the standard encoder based on the paper Vaswani (2017). We utilize the Transformer encoder layer from the PyTorch library, which comprises a self-attention mechanism and a feedforward network. Unlike Jeong et al. (2023), we did not implement their custom relative positional encoding, as our focus is solely on employing the Transformer model as an encoding network.

2.3 Dependency Modeling with Copulas in Embedding Space

A copula models how the embedding dimensions $(z_1, \ldots, z_d)$ co-vary, focusing on their dependency structure irrespective of individual marginal distributions. If normal embeddings $\mathbf{z}$ maintain certain correlational patterns (e.g., z_1 rises when z_3 drops), the copula assigns them high log-density. Anomalies produce embeddings that violate these learned dependencies, thus yielding lower copula log-density.

The Transformer encoder ensures each embedding or latent vector $\mathbf{z}$ is an expressive representation of temporal and cross-feature relationships of the original time series. Meanwhile, the copula provides a precise measure of *joint* likelihood for those latent coordinates. Consequently, normal data reside in a dense, high-likelihood region of embedding space, whereas anomaly embeddings occupy a lower-likelihood, lower-density region. This synergy leads to a more robust separation than simpler, local methods that may overlook global time dependencies.

2.4 Contrastive Loss

We model the joint distribution of latent variables $\mathbf{z} = (z_1, \ldots, z_d)$ using a copula $c_\phi(\mathbf{z})$, which effectively captures their underlying dependencies irrespective of marginal distributions, and the *log-density* $\log c_\phi(\mathbf{z})$ can be interpreted as reflecting the *mutual information* among these variables. If $\mathbf{z}$ aligns with the normal dependency structure, $c_\phi(\mathbf{z})$ is high (i.e., strong correlations), whereas anomalies that break these dependencies yield a low copula log-density. Indeed, the copula log-likelihood is the sum of $\log c_\phi$ over samples; thus, maximizing copula likelihood is equivalent to maximizing the multi-dimensional dependency, or high MI, in the embedding space. To exploit this for anomaly detection, we construct a *contrastive loss* that (1) maximizes $\log c_\phi\big(\mathbf{z}_n(\theta)\big)$ for normal samples $\mathbf{z}_n$, ensuring that normal data inhabit a high-MI region, and (2) forces anomalous samples $\mathbf{z}_a$ into a lower-density region.

Specifically, we jointly optimize the *encoder parameters* θ (mapping raw time series to embedding space $\mathbf{z}$) and *copula parameters* ϕ by minimizing the loss function

$$\begin{aligned}\mathcal{L}(\theta, \phi) = &- \sum_{n \in \text{Norm}} \log c_\phi\big(\mathbf{z}_n(\theta)\big) + \\ &\alpha \sum_{a \in \text{Anom}} \max\{0, \log\, c_\phi(\mathbf{z}_a(\theta)) - (\mu_{\text{norm}} - \delta)\},\end{aligned} \tag{1}$$

where Norm and Anom are normal and anomaly data frames.

A common concern arises because the copula log-density, $\log c_\phi(\mathbf{z})$, is often negative for high-dimensional data, so *maximizing* this quantity corresponds to *minimizing* its negative. Hence, for normal samples we add the term $-\log c_\phi\big(\mathbf{z}_n(\theta)\big)$ in the loss, ensuring that we push those log-densities *toward zero* (i.e., less negative). To separate anomalies, we place a margin constraint that the anomalous log-density stays below $\mu_{\text{norm}} - \delta$, where $\mu_{\text{norm}} < 0$ is the average log-density over normal data and $\delta > 0$ is a margin. We then penalize any anomaly whose log-density $\log c_\phi(\mathbf{z}_a(\theta))$ exceeds $(\mu_{\text{norm}} - \delta)$. Backpropagating with respect to both encoder (θ) and copula (ϕ) parameters end-to-end guides the model to learn the embeddings that preserve key time-dependent correlations for normal data (thereby raising their mutual information), while anomaly embeddings deviate and incur a higher penalty.

2.5 Synthetic Data Generation Scheme

Since our training data have only normal data, we generate degraded inputs by replacing parts of a window with outliers during the training phase. Similarly to Anomaly-BERT Jeong et al. (2023), we randomly select an interval $[t_0, t_1]$ within a window $\mathbf{X} = x_{t_0:t_1}$. The selected sequence $\mathbf{X} = x_{t_0:t_1}$ is then replaced by one of the synthetic outliers described in the following.

2.5.1 Local Perturbation

We introduce a local perturbation scheme to build on the degradation methods proposed by Jeong et al. (2023). Local perturbation generates synthetic anomalies by introducing small controlled modifications to real anomaly samples. The steps are as follows:

- Randomly select a real anomaly snippet from the reference anomaly dataset.
- Adjust the snippet's size to match the target window length by clamping or padding.

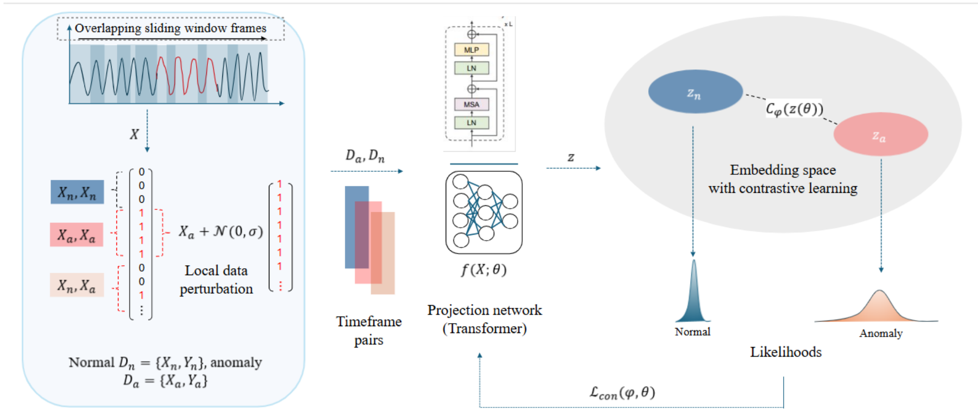


Figure 1: Spatio-Temporal Dependency Aware Feature Learning Framework

- Apply small perturbations, such as:
  - Adding Gaussian noise proportional to the snippet's standard deviation.
  - Scaling features slightly using random scaling factors in a small range (e.g., $[0.95, 1.05]$).
  - Permuting a small subset of rows to introduce mild temporal variations.
- Overlay the perturbed snippet onto the target sequence.

This approach ensures that the generated anomalies remain close to the real anomaly distribution while introducing variability for robustness. For the local perturbation approach, we utilize 10-15% of the labeled anomalies to create new anomalies that closely mimic the characteristics of true anomalies.

3 Experiments and Results

In this section, we begin by outlining the experimental setup. Following that, we conduct a series of experiments to evaluate the effectiveness of the model, classification results, and the results of ablation studies.

3.1 Datasets and Baseline Methods

We present experimental results on five widely-used benchmark datasets: SWaT, WADI, SMAP, MSL, and SMD Goh et al. (2017), Ahmed et al. (2017), Hundman et al. (2018), Su et al. (2019). These datasets are derived from various sources, including sensors in server machines, spacecraft, and water treatment or distribution systems. We use a small portion of the labeled anomalies (10-15 %) approximately to generate synthetic anomalies with our local perturbation scheme. Again, we use only the continuous features in all the datasets for modeling. Several advanced models have been proposed for anomaly detection in multivariate time series. MERLIN Nakamura et al. (2020) is a self-supervised method that generates pseudo-labels by learning representations and applies contrastive learning to detect anomalies effectively. LSTM-NDT Hundman et al. (2018) leverages LSTM networks for neural density estimation, capturing temporal dependencies in multivariate time series. DAGMM Zong et al. (2018) combines dimensionality reduction with Gaussian mixture models through a deep autoencoding framework to estimate density and identify anomalies. OmniAnomaly Su et al. (2019) employs a variational RNN-based architecture to model temporal dependencies and reconstruct inputs using stochastic latent variables for anomaly detection. MSCRED Zhang et al. (2019) reconstructs multi-scale signature matrices via a convolutional recurrent encoder-decoder to capture temporal correlations and detect anomalies. MAD-GAN Li et al. (2019) applies an adversarial framework using GANs to reconstruct normal patterns, flagging significant reconstruction deviations as anomalies. USAD Audibert et al. (2020) integrates adversarial training with autoencoders in a unified framework to learn patterns and identify anomalies. MTAD-GAT Zhao et al. (2020) utilizes graph attention networks to effectively capture spatial and temporal dependencies in multivariate time series. CAE-M

Zhang et al. (2021) applies a convolutional autoencoder to reconstruct and predict temporal dependencies, facilitating anomaly detection. GDN Deng & Hooi (2021) leverages graph neural networks to model inter-variable dependencies, enhancing anomaly detection performance. Finally, TranAD Tuli et al. (2022) adopts a transformer-based architecture with attention mechanisms to model long-term dependencies and effectively identify anomalies in multivariate time series. We also implement CARLA Darban et al. (2025) and Anomaly-Bert Jeong et al. (2023) as baseline models.

Table 1: Performance metrics (Precision(P), Recall(R), AUC, F1) for various methods across datasets.

Method	WADI				SWaT				MSL			
	P	R	AUC	F1	P	R	AUC	F1	P	R	AUC	F1
MERLIN	0.0636	0.7669	0.5912	0.1174	0.6560	0.2547	0.6175	0.3669	0.2613	0.4645	0.6281	0.3345
LSTM-NDT	0.0138	0.7823	0.6721	0.0271	0.7778	0.5109	0.7140	0.6167	0.6288	1.0000	0.9532	0.7721
DAGMM	0.0760	0.9981	0.8563	0.1412	0.9933	0.6879	0.8436	0.8128	0.7363	1.0000	0.9716	0.8482
OmniAnomaly	0.3158	0.6541	0.8198	0.4260	0.9782	0.6957	0.8467	0.8131	0.7848	0.9924	0.9782	0.8765
MSCRED	0.2513	0.7319	0.8412	0.3741	**0.9992**	0.6770	0.8433	0.8072	0.8912	0.9862	0.9807	0.9363
MAD-GAN	0.2233	0.9124	0.8026	0.3588	0.9593	0.6957	0.8463	0.8065	0.8516	0.9930	0.9862	0.9169
USAD	0.1873	0.8296	0.8723	0.3056	0.9977	0.6879	0.8460	0.8143	0.7949	0.9912	0.9795	0.8822
MTAD-GAT	0.2818	0.8012	0.8821	0.4169	0.9718	0.6957	0.8464	0.8109	0.7917	0.9824	0.9899	0.8768
CAE-M	0.2782	0.7918	0.8728	0.4117	0.9697	0.6957	0.8464	0.8101	0.7751	1.0000	0.9903	0.8733
GDN	0.2912	0.7931	0.8777	0.4260	0.9697	0.6957	0.8464	0.8101	0.9308	0.9892	0.9814	0.9591
TranAD	**0.3529**	**0.8296**	**0.8968**	0.4951	0.9760	0.6997	0.8491	0.8151	0.9038	**0.9999**	0.9916	0.9494
Anomaly-Tran	NA	NA	NA	NA	0.9155	0.9673	NA	0.9407	0.9209	0.9515	NA	0.9359
Anomaly-Bert	NA	NA	NA	**0.5800**	NA	NA	NA	0.8540	NA	NA	NA	0.3020
CARLA	0.1850	0.7316	NA	0.2953	0.9886	0.5673	NA	0.7209	0.3891	0.7959	NA	0.5227
Our Model	0.2776	0.7912	0.7254	0.4110	0.9907	**0.9969**	**0.9999**	**0.9979**	**0.9397**	0.9956	**0.9999**	**0.9668**

Method	SMD				SMAP			
	P	R	AUC	F1	P	R	AUC	F1
MERLIN	0.2871	0.5804	0.7158	0.3842	0.1577	0.9999	0.7426	0.2725
LSTM-NDT	0.9736	0.8440	0.9671	0.9042	0.8523	0.7326	0.8602	0.7879
DAGMM	0.9103	0.9914	0.9954	0.9491	0.8069	0.9891	0.9885	0.8888
OmniAnomaly	0.8881	0.9985	0.9946	0.9401	0.8130	0.9419	0.9889	0.8728
MSCRED	0.7276	0.9974	0.9921	0.8414	0.8175	0.9216	0.9821	0.8664
MAD-GAN	0.9991	0.8440	0.9933	0.9150	0.8157	0.9216	0.9891	0.8915
USAD	0.9060	0.9974	0.9933	0.9495	0.7480	0.9627	0.9890	0.8419
MTAD-GAT	0.8210	0.9215	0.9921	0.8683	0.7991	0.9991	0.9844	0.8880
CAE-M	0.9082	0.9671	0.9783	0.9367	0.8193	0.9567	0.9901	0.8827
GDN	0.7170	0.9974	0.9924	0.8342	0.8195	0.9312	**0.9981**	0.8695
TranAD	0.9262	0.9974	**0.9974**	0.9605	0.7480	0.9891	0.9864	0.8518
Anomaly-Tran	0.8940	0.9545	NA	0.9233	0.9413	0.9940	NA	0.9669
Anomaly-Bert	NA	NA	NA	0.535	NA	NA	NA	0.4570
CARLA	0.4276	0.6362	NA	0.5114	0.3944	0.8040	NA	0.5292
Our Model	**0.9751**	**0.9998**	0.9934	**0.9873**	**0.9795**	**0.9923**	0.9841	**0.9859**

3.2 Training Strategy

We train our network by jointly optimizing the Transformer encoder parameters, denoted θ, and the copula parameters, denoted ϕ, in a single end-to-end fashion. First, each multivariate time series snippet is passed through the Transformer encoder, which produces a latent embedding or feature space $\mathbf{z}(\theta)$ capturing relevant temporal and cross-feature dependencies. We then evaluate the copula log-density $\log c_\phi\big(\mathbf{z}(\theta)\big)$ to quantify how well the latent feature space aligns with the joint dependency structure learned from normal data. During training, we formulate a contrastive loss that *maximizes* $\log c_\phi\big(\mathbf{z}(\theta)\big)$ for normal frames while *pushing* anomalous frames below a specified margin in log-density space. Specifically, for each normal snippet we minimize $-\log c_\phi\big(\mathbf{z}(\theta)\big)$, thus maximizing the copula likelihood, and for each anomalous snippet we include a term that penalizes its log-density if it is not sufficiently lower than the normal average. By backpropagating through both the Transformer and the copula parameters, the system iteratively updates θ and ϕ such that normal data cluster in a high-likelihood region, while anomalies are assigned lower density and thus become well separated in the latent feature space. Figure 1 presents a schematic illustration of our model.

3.3 Hyperparameters

The following are the key hyperparameters on which our model performance depends.

Window Size (L): Each multivariate time-series snippet is of length L, meaning we process L consecutive timestamps as a single input frame. This choice controls how much local context the model sees at once.
Overlap or Step Length: After extracting a window of length L, we often shift by a smaller step (e.g., $L/2$ or another fraction) for the next window, creating overlapping frames. Overlaps help capture transitions more smoothly and increase data availability, but can lead to redundancy if the overlap is too large.
Number (or Percentage) of Anomalies: We typically define what fraction (e.g., 10% or 30%) of frames to label as anomalous for synthetically generated sets. We gradually vary this percentage and check the model performance with various percentages of synthetic anomaly samples.
Batch Size (B): The batch size indicates how many frames we process in one forward/backward pass.
Transformer Depth and Heads: Although not strictly a hyperparameter, we experiment with different numbers of layers, attention heads, and embedding dimensions.
Copula Family and Parameters: In the latent feature space, we choose a specific copula family (Gaussian, Student-t) along with its parameters or estimation method. This affects how well the model captures the joint dependency structure among latent feature dimensions.
Margin (δ) in Contrastive Loss: We define a margin δ that ensures that anomalies remain sufficiently below the normal log-likelihood region. Larger δ forces stricter separation but can cause more false alarms if the model over-penalizes borderline samples. We employ a sample-level margin, ensuring that each anomaly sample lies below that margin.
Learning Rate and Optimizer: Finally, we choose an optimizer (e.g., Adam) and a learning rate for both the Transformer encoder parameters and the copula parameters. Tuning this is crucial to ensure stable, efficient convergence.
Weightage (α): We define the α parameter to balance the different components of our loss function. We experiment with different values of α.

3.4 Copula-Based Modeling for Latent-Space Dependency

In our anomaly detection framework, the latent representation $\mathbf{z} \in \mathbb{R}^d$ captured by a Transformer encoder undergoes *copula modeling* to quantify the joint dependency among the dimensions of $\mathbf{z}$. We focus on two main copula families: the *Gaussian copula* and the *Student-t copula*. Each of these approaches primarily learns a *correlation structure*, denoted by ϕ, in the latent space, allowing us to evaluate how well a given $\mathbf{z}$ aligns with typical (normal) behavior.

Algorithm 1 Gaussian Copula for Latent Dependencies

Input : Latent vector $\mathbf{z} \in \mathbb{R}^d$, Cholesky parameters ϕ
Output: Log-likelihood $\log c_{\text{Gauss}}(\mathbf{z};\phi)$

1. Standardise to zero mean and unit variance $\mathbf{z}_{\text{std}} \leftarrow (\mathbf{z}-\mu)/\sigma$
2. Transform to $(0,1)$ via Normal CDF $u_i \leftarrow \Phi\big(z_{\text{std},i}\big)$
3. Map back using inverse Normal CDF $z_i^{(\text{transf})} \leftarrow \Phi^{-1}(u_i)$
4. Build correlation $\Sigma = LL^\top$; $L \leftarrow$ Cholesky factor from params(ϕ); $\Sigma \leftarrow L\,L^\top$
5. Multivariate Gaussian log-likelihood $\log c_{\text{Gauss}} \leftarrow -\frac{1}{2}\left(\left(\mathbf{z}^{(\text{transf})}\right)^\top \Sigma^{-1}\mathbf{z}^{(\text{transf})} + \log\det\Sigma + d\log 2\pi\right)$

return $\log c_{Gauss}$

Since we concentrate on the *correlation parameters* ϕ (which define L), our optimization updates only those elements to best fit normal data. Anomalies are detected if they yield low log-likelihood under this Gaussian correlation structure. The Gaussian copula is simple and captures linear correlation well. It is suitable when tail dependence is not extreme, or when anomalies primarily break moderate cross-variable correlations.

However, if the data exhibit heavier tails or outliers, the Gaussian copula may underestimate such tail dependencies.

3.5 Student-t Copula for Heavy-tailed Dependencies

To address more pronounced tail behavior, we can replace the Gaussian assumption with a **Student-t copula**, which has an extra *degrees of freedom* (ν) parameter governing tail thickness. Concretely,

Algorithm 2 Student-t Copula for Heavy-Tailed Dependencies

Input : Latent vector $\mathbf{z} \in \mathbb{R}^d$, degrees-of-freedom $\nu > 0$, Cholesky params ϕ
Output: Log-likelihood $\log c_t(\mathbf{z}; \phi, \nu)$

1. Standardise as in Gaussian case $\mathbf{z}_{\text{std}} \leftarrow (\mathbf{z} - \mu)/\sigma$
2. Transform to $(0, 1)$ via t-CDF $u_i \leftarrow T_\nu\big(z_{\text{std},i}\big)$
3. Map back by inverse t-CDF $z_i^{(\text{transf})} \leftarrow T_\nu^{-1}(u_i)$
4. Build correlation $\Sigma = LL^\top$; $L \leftarrow$ Cholesky factor from params(ϕ) ; $\Sigma \leftarrow L\,L^\top$
5. Multivariate Student-t log-likelihood $\log c_t \leftarrow \log\Gamma\big(\frac{\nu+d}{2}\big) - \log\Gamma\big(\frac{\nu}{2}\big) - \frac{1}{2}\log\det(\nu\pi\,\Sigma) - \frac{\nu+d}{2}\log\Big(1 + \frac{1}{\nu}\big(\mathbf{z}^{(\text{transf})}\big)^\top\Sigma^{-1}\mathbf{z}^{(\text{transf})}\Big)$

return $\log c_t$

This approach again optimizes primarily the *correlation parameters* ϕ (related to L) *and* we may also learn or fix the ν parameter. Hence, anomalies that yield *unexpected* tail dependencies are flagged with low likelihood. Real-world signals often exhibit outliers or heavy-tailed distributions. A Student-t copula accommodates such extremes more naturally than the Gaussian copula, assigning higher probability mass in the tails. Thus, anomalies deviating in a heavy-tailed manner are more cleanly separated. If data indeed show large spikes, a t-copula typically yields more robust anomaly detection than its Gaussian counterpart.

3.5.1 Learning the Correlation Parameter ϕ

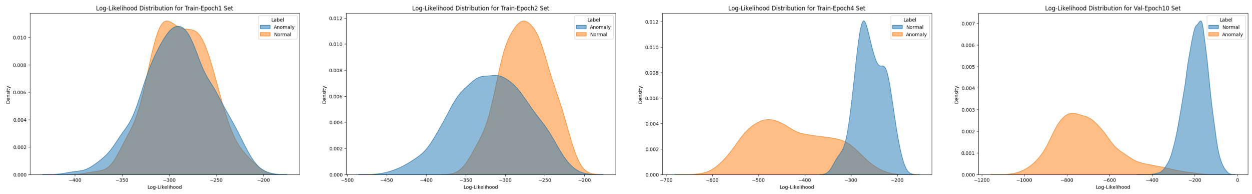
In both copula families, our primary focus is on *learning the correlation structure* that defines how the latent dimensions co-vary for normal samples. Concretely, we store a vector of Cholesky parameters ϕ and the parameter ν in the Student-t case, and backprop through these when fitting normal data in training. At inference, any $\mathbf{z}$ that fails to match this learned dependency pattern receives a lower log-likelihood and is deemed more likely anomalous.

By choosing either a *Gaussian* or *Student-t* copula for latent-space modeling, we gain flexibility in how tail dependencies are captured. The Gaussian copula is often simpler and sufficient for moderate dependencies, whereas the Student-t copula provides a heavier-tailed alternative that can handle more extreme data, thus yielding improved anomaly detection in tail-heavy scenarios. Throughout, we primarily tune the correlation parameters ϕ, enabling an end-to-end training scheme where the latent encoder (e.g. a Transformer) supplies embedded features, and the copula measures how well they align with the normative correlation structure.

3.6 Ablation Study

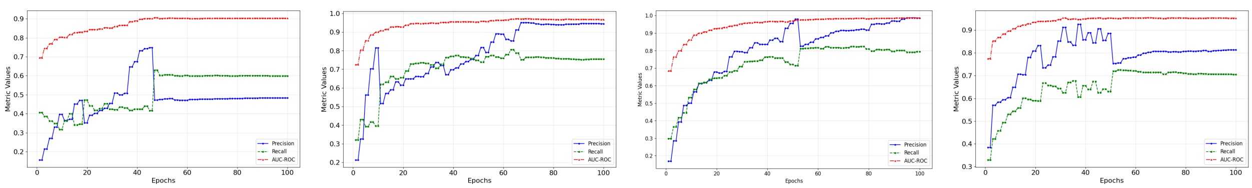
In our ablation study, we systematically vary several hyperparameters to observe their impact on anomaly detection performance. First, we adjust the *margin* δ in our contrastive loss, finding that too small a margin may allow anomaly frames to encroach on normal regions, while too large a margin can over-penalize borderline anomalies and lead to increased false positives. We also vary the *percentage of anomaly data* we inject or label during training, confirming that a higher anomaly fraction generally helps the model to better distinguish anomalies, although an unrealistically high percentage may degrade generalization to real, rarer anomalies.

Furthermore, we experiment with different copula families in the latent feature space—for instance, a *Gaussian* copula, a *Student-t* copula (better for heavy-tailed dependencies). We also vary *window size* and *overlap*



(a) *epoch* = 1 (b) *epoch* = 5 (c) *epoch* = 20 (d) *epoch* = 25

Figure 2: We jointly train the transformer and copula (student-t) parameters in the latent space with the contrastive loss and plot the normal and anomaly likelihoods. We see the gradual separation in likelihoods as we train for more epochs using our method.



(a) δ = 300, (25%) (b) δ = 800, (25%) (c) δ = 500, (30%) (d) δ = 500, (35%)

Figure 3: Performance metrics (Precision, recall, AUC-ROC) over epochs on the validation data on the SWaT dataset. The hyper-parameters margin δ and the percentage of anomaly frames (in parenthesis) are varied for the best performance. Here, we use the Gaussian copula for the latent space dependency modeling.

in the construction of time-series frames, observing that larger windows capture extended patterns but raise computational cost, while more overlap provides smoother coverage but increases data redundancy. Finally, we alter the *Transformer encoder architecture* by changing the number of layers and attention heads. In Figure 3, we plot the precision, recall, AUC-ROC on the validation data during each iteration of the training for different values of the hyperparameters margin and percentage of anomaly frames. We find the best model performance when the margin (δ) = 500 and the percentage of anomaly frames introduced is 35% on the SWaT dataset. Other results are provided in Table 1. In Figure 2, we observe that as training progresses, the model achieves increasingly better separation between the likelihoods of the normal and anomalous instances in the latent space. [1]

3.7 Anomaly Detection Threshold Selection

In this approach, we first derive a *log-likelihood score* for each time series frame by evaluating $\log c_\phi(\mathbf{z}(\theta))$, where c_ϕ is the copula density function and $\mathbf{z}(\theta)$ is the latent embedding returned by our Transformer encoder. Intuitively, a frame with higher (less negative) log-likelihood aligns better with normal behavior, while lower log-likelihood indicates a potential anomaly. We gather all log-likelihoods from the validation set and systematically scan a range of possible thresholds, from the minimum to the maximum observed score. For each candidate threshold τ, we label a frame as anomalous if its log-likelihood is below τ, and we compute the corresponding F1 score on the validation data. We then select the threshold that *maximizes* the F1, thus balancing precision and recall most effectively. Once this threshold is determined, any future frame whose log-likelihood drops below τ is deemed anomalous, while frames exceeding τ are considered normal.

4 Gradient Computation and Backpropagation

We focus on computing gradients of the contrastive loss with respect to ϕ (the copula parameters that capture joint dependency) and θ (the Transformer encoder parameters that generate latent embeddings). Here, each *frame*—a short subsequence of the time-series—is treated as an individual item, potentially labeled 0 (normal) or 1 (anomalous).

[1] `https://anonymous.4open.science/r/DACLM-7152/`

Algorithm 3 Single–batch update of Transformer encoder θ and Copula parameters ϕ

Input : Mini-batch $\mathcal{B} = \{(x^{(i)}, y^{(i)})\}_{i=1}^{N}$ with $y^{(i)} \in \{0, 1\}$
Output: Updated parameters θ, ϕ

1. Initialise learning rates η, margin γ, and standardisation parameters (μ, σ)
2. Forward pass over mini-batch For each $i = 1, \ldots, N$:
   - Encode: $\mathbf{z}^{(i)} \leftarrow f_{\theta}^{\text{enc}}(x^{(i)})$
   - Standardise: $\mathbf{z}_{\text{std}}^{(i)} \leftarrow (\mathbf{z}^{(i)} - \mu)/\sigma$
   - Transform each component: $u_k^{(i)} \leftarrow G_k(z_{\text{std},k}^{(i)})$ for $k = 1, \ldots, d$
   - Compute loss: $\ell^{(i)} \leftarrow -\log c(\mathbf{u}^{(i)}, \phi)$
   - If $y^{(i)} = 1$, apply contrastive margin: $\ell^{(i)} \leftarrow \ell^{(i)} + \gamma$
3. Average total loss $\mathcal{L}(\theta, \phi) \leftarrow \frac{1}{N} \sum_{i=1}^{N} \ell^{(i)}$
4. Backpropagation
   - Copula gradient: $\nabla_{\phi}\mathcal{L} = -\frac{1}{N} \sum_{i=1}^{N} \frac{\partial}{\partial \phi} \log c(\mathbf{u}^{(i)}, \phi)$
   - Encoder gradient: $\nabla_{\theta}\mathcal{L} = \sum_{i=1}^{N} \frac{\partial \mathcal{L}}{\partial \mathbf{z}^{(i)}} \cdot \frac{\partial \mathbf{z}^{(i)}}{\partial \theta}$
5. Parameter updates $\theta \leftarrow \theta - \eta\nabla_{\theta}\mathcal{L}$; $\phi \leftarrow \phi - \eta\nabla_{\phi}\mathcal{L}$

return θ, ϕ

By treating entire frames rather than individual timesteps, we more naturally capture local temporal context, while the joint optimization of (θ, ϕ) encourages both a suitable embedding space and an accurate copula-based dependency model. Anomalies emerge as frames that fail to fit this latent dependency pattern, receiving lower log-likelihood under $c(\mathbf{u}, \phi)$ and thus incurring higher contrastive penalty.

By defining a contrastive loss $\mathcal{L}(\theta, \phi)$ that rewards high log-likelihood for normal data and penalizes anomalies, we can backpropagate through both the *latent mapping* (i.e. Transformer encoder) and the *copula model* to learn parameters $\{\theta, \phi\}$. In essence:

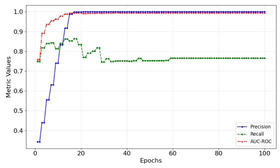
$$\theta \longleftarrow \theta - \eta \nabla_{\theta}\mathcal{L}(\theta, \phi), \quad \phi \longleftarrow \phi - \eta \nabla_{\phi}\mathcal{L}(\theta, \phi),$$

where η is the learning rate. Hence, the system adaptively modifies both the latent representations and the copula-based dependency structure to differentiate normal from anomalous samples in the most effective way.

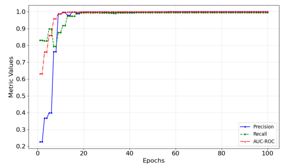
5 Hyperparameter Tuning and Results

In the below section, we experiment with the different values of the hyperparameters and validate our model performance. In Figure 4 our experiments reveal that the classification performance of our model is significantly influenced by the selected copula family used to capture the joint dependency of the latent variables. This finding suggests that the Student-t copula provides a more accurate representation of the data-generating process, characterized by heavy-tailed distributions, rather than the Gaussian distribution. We keep the other hyper-parameters like margin, batch-size, window-length fixed for all the cases.

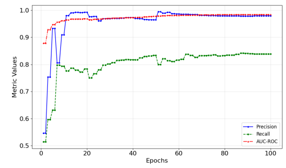
Our model achieves, on average, a 5–18% improvement in classification metrics over baseline methods on most datasets. With the WADI dataset, the performance degradation is likely due to an exception to our assumption that variables exhibit distinct joint dependency structures under normal and anomalous conditions. We also observe poor performance of other baselines on the same dataset. Through ablation in Figure 4, we show that incorporating a Student-t copula—capable of modeling heavy-tailed and non-linear



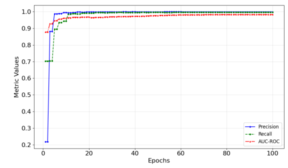
(a) Latent Space with Gaussian copula (MSL)



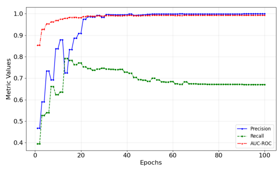
(b) Latent Space with Student-t copula (MSL)



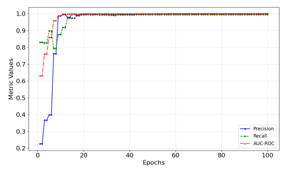
(c) Latent Space with Gaussian copula (SMAP)



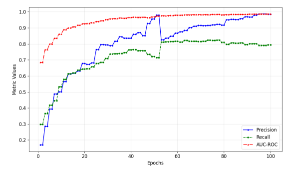
(d) Latent Space with Student-t copula (SMAP)



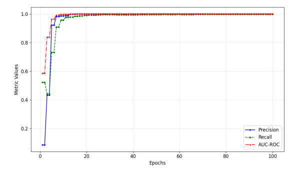
(e) Latent Space with Gaussian copula (SMD)



(f) Latent Space with Student-t copula (SMD)



(g) Latent Space with Gaussian copula (SWAT)



(h) Latent Space with Student-t copula (SWAT)

Figure 4: Performance metrics (Precision, recall, AUC-ROC) over epochs on the validation data when the latent space is modeled with Gaussian Copula and Student-t Copula for different datasets.

dependencies—leads to significant gains. Distributional analysis confirms that the feature marginals are highly skewed and deviate from Gaussianity, necessitating more expressive copula models.

Computation Complexity and high-dimensional setting We implement Gaussian and Student-t copulas in a low-dimensional latent space by parameterizing the correlation matrix $\Sigma = LL^T$ using its Cholesky factor L, with positive diagonals enforced via a `softplus` function. Latent variables are standardized, mapped through the Student-t CDF, and transformed back using the inverse CDF (PPF). The log-determinant is computed as $\log|\Sigma| = 2\sum \log(L_{ii})$ in $O(d)$ time, and the quadratic form $x^T\Sigma^{-1}x$ is evaluated via einsum operations with $O(d^2)$ complexity. Although Cholesky inversion has a worst-case cost of $O(d^3)$, the small latent dimension d makes these operations computationally efficient. In our case, with datasets having fewer than 100 features, the added overhead is minimal. For high-dimensional settings, Salinas et al. (2019) propose decomposing the covariance matrix into a diagonal and a low-rank component, reducing parameters from $O(N^2)$ to $O(Nr)$, and overall complexity to $O(Nr^2)$, which is effectively $O(N)$ for small fixed r.

6 Conclusion:

In our paper, we have created a unified, end-to-end anomaly detection framework that captures both temporal and multivariate relationships in complex time series. The joint modeling helps to capture the complex time-varying spatio-temporal non-linear correlations which are useful indicators of multi-variate anomalies. This approach yields a latent representation guided by copula likelihoods, effectively separating normal frames (high likelihood) from anomalous ones (low likelihood). Future work may explore more efficient attention mechanisms, advanced mixture copulas for even richer tail behaviors to improve anomaly detection in the latent space.

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*, pp. 25–28, 2017.

Samet Akcay, Amir Atapour-Abarghouei, and Toby P Breckon. Ganomaly: Semi-supervised anomaly detection via adversarial training. In *Computer Vision–ACCV 2018: 14th Asian Conference on Computer*

Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14, pp. 622–637. Springer, 2019.

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*, pp. 3395–3404, 2020.

Mikel Canizo, Isaac Triguero, Angel Conde, and Enrique Onieva. Multi-head cnn–rnn for multi-time series anomaly detection: An industrial case study. *Neurocomputing*, 363:246–260, 2019.

Andrew A Cook, Göksel Mısırlı, and Zhong Fan. Anomaly detection for iot time-series data: A survey. *IEEE Internet of Things Journal*, 7(7):6481–6494, 2019.

Zahra Zamanzadeh Darban, Geoffrey I Webb, Shirui Pan, Charu C Aggarwal, and Mahsa Salehi. Carla: Self-supervised contrastive representation learning for time series anomaly detection. *Pattern Recognition*, 157:110874, 2025.

Ailin Deng and Bryan Hooi. Graph neural network-based anomaly detection in multivariate time series. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pp. 4027–4035, 2021.

Alexandre Drouin, Étienne Marcotte, and Nicolas Chapados. Tactis: Transformer-attentional copulas for time series. In *International Conference on Machine Learning*, pp. 5447–5493. PMLR, 2022.

Jonathan Goh, Sridhar Adepu, Khurum Nazir Junejo, and Aditya Mathur. A dataset to support research in the design of secure water treatment systems. In *Critical Information Infrastructures Security: 11th International Conference, CRITIS 2016, Paris, France, October 10–12, 2016, Revised Selected Papers 11*, pp. 88–99. Springer, 2017.

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*, pp. 387–395, 2018.

Yungi Jeong, Eunseok Yang, Jung Hyun Ryu, Imseong Park, and Myungjoo Kang. Anomalybert: Self-supervised transformer for time series anomaly detection using data degradation scheme. *arXiv preprint arXiv:2305.04468*, 2023.

Watson Jia, Raj Mani Shukla, and Shamik Sengupta. Anomaly detection using supervised learning and multiple statistical methods. In *2019 18th IEEE international conference on machine learning and applications (ICMLA)*, pp. 1291–1297. IEEE, 2019.

Dan Li, Dacheng Chen, Baihong Jin, Lei Shi, Jonathan Goh, and See-Kiong Ng. Mad-gan: Multivariate anomaly detection for time series data with generative adversarial networks. In *International conference on artificial neural networks*, pp. 703–716. Springer, 2019.

Takaaki Nakamura, Makoto Imamura, Ryan Mercer, and Eamonn Keogh. Merlin: Parameter-free discovery of arbitrary length anomalies in massive time series archives. In *2020 IEEE international conference on data mining (ICDM)*, pp. 1190–1195. IEEE, 2020.

David Salinas, Michael Bohlke-Schneider, Laurent Callot, Roberto Medico, and Jan Gasthaus. High-dimensional multivariate forecasting with low-rank gaussian copula processes. *Advances in neural information processing systems*, 32, 2019.

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*, pp. 2828–2837, 2019.

Liaoyuan Tang, Zheng Wang, Guanxiong He, Rong Wang, and Feiping Nie. Perturbation guiding contrastive representation learning for time series anomaly detection. In *Proc. 33rd Int. Joint Conf. Artif. Intell*, pp. 4955–4963, 2024.

Markus Thill, Wolfgang Konen, Hao Wang, and Thomas Bäck. Temporal convolutional autoencoder for unsupervised anomaly detection in time series. *Applied Soft Computing*, 112:107751, 2021.

Zhiwen Tian, Ming Zhuo, Leyuan Liu, Junyi Chen, and Shijie Zhou. Anomaly detection using spatial and temporal information in multivariate time series. *Scientific Reports*, 13(1):4400, 2023.

Shreshth Tuli, Giuliano Casale, and Nicholas R Jennings. Tranad: Deep transformer networks for anomaly detection in multivariate time series data. *arXiv preprint arXiv:2201.07284*, 2022.

A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

Chaoyang Wang and Guangyu Liu. From anomaly detection to classification with graph attention and transformer for multivariate time series. *Advanced Engineering Informatics*, 60:102357, 2024.

Jiehui Xu. Anomaly transformer: Time series anomaly detection with association discrepancy. *arXiv preprint arXiv:2110.02642*, 2021.

Zhaoping Xu, Zhijun Cheng, and Bo Guo. A multivariate anomaly detector for satellite telemetry data using temporal attention-based lstm autoencoder. *IEEE Transactions on Instrumentation and Measurement*, 2023.

Chuxu Zhang, Dongjin Song, Yuncong Chen, Xinyang Feng, Cristian Lumezanu, Wei Cheng, Jingchao Ni, Bo Zong, Haifeng Chen, and Nitesh V Chawla. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 1409–1416, 2019.

Yuxin Zhang, Yiqiang Chen, Jindong Wang, and Zhiwen Pan. Unsupervised deep anomaly detection for multi-sensor time-series signals. *IEEE Transactions on Knowledge and Data Engineering*, 35(2):2118–2132, 2021.

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 *2020 IEEE international conference on data mining (ICDM)*, pp. 841–850. IEEE, 2020.

Bo Zong, Qi Song, Martin Renqiang Min, Wei Cheng, Cristian Lumezanu, Daeki Cho, and Haifeng Chen. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In *International conference on learning representations*, 2018.

A Appendix

Copula Theory

1. Copula Function: Copulas allow us to represent the joint distribution of random variables using their marginal distributions and a copula function that captures their dependency structure. The joint PDF can be written as:

$$p(\mathbf{z}) = c(\mathbf{u}; \phi) \prod_{i=1}^{d} p(z_i),$$

where $\mathbf{u} = (u_1, u_2, \dots, u_d)$ with $u_i = F_{Z_i}(z_i)$ (the marginal CDF of Z_i), and $c(\mathbf{u}; \phi)$ is the copula density.

2. Dependency Structure: The copula density $c(\mathbf{u}; \phi)$ encapsulates the dependency structure among the variables, while the marginal distributions $p(z_i)$ account for their individual behaviors.

Rewriting Mutual Information Using Copulas

Substituting the copula representation of $p(\mathbf{z})$ into the mutual information formula:

$$I(\mathbf{Z}) = \int p(\mathbf{z}) \log \left(\frac{c(\mathbf{u}; \phi) \prod_{i=1}^{d} p(z_i)}{\prod_{i=1}^{d} p(z_i)} \right) d\mathbf{z}.$$

Simplifying the logarithmic term:

$$I(\mathbf{Z}) = \int p(\mathbf{z}) \log c(\mathbf{u}; \phi)\, d\mathbf{z}.$$

Change of Variables From $\mathbf{z}$ to $\mathbf{u}$

To simplify the integral, we perform a change of variables:

$$u_i = F_{Z_i}(z_i), \quad z_i = F_{Z_i}^{-1}(u_i).$$

The Jacobian determinant of this transformation is given by:

$$|J| = \prod_{i=1}^{d} p(z_i).$$

Thus, the volume element transforms as:

$$d\mathbf{z} = \frac{d\mathbf{u}}{|J|} = \frac{d\mathbf{u}}{\prod_{i=1}^{d} p(z_i)}.$$

Substituting this change of variables into the integral:

$$I(\mathbf{Z}) = \int c(\mathbf{u}; \phi) \log c(\mathbf{u}; \phi)\, d\mathbf{u}.$$

Final Expression for Mutual Information

The mutual information among the variables $\mathbf{Z}$ is expressed in terms of the copula density as:

$$I(\mathbf{Z}) = \int c(\mathbf{u}; \phi) \log c(\mathbf{u}; \phi)\, d\mathbf{u}.$$

This formulation shows that mutual information is equivalent to the expected log-likelihood of the copula density. The copula density $c(\mathbf{u}; \phi)$ captures the dependency structure among the variables, independent of their marginal distributions, providing a comprehensive measure of dependency.

B Transformer Encoder Configurations

Although we explored various Transformer encoder configurations, we did not observe a direct improvement in classification performance with increased model complexity, such as adding more layers or attention heads. Interestingly, a lightweight model (e.g., Config 1, Config 2) achieved the best performance in our experiments.

Table 2: Detailed Configurations of Transformer Encoder Models

Parameter	Config 1	Config 2	Config 3	Config 4
Input Dimension (input_dim)	32	64	32	128
Model Dimension (model_dim)	64	128	256	512
Number of Layers (num_layers)	4	6	8	12
Number of Attention Heads (num_heads)	2	4	8	8
Dropout Rate (dropout)	0.1	0.2	0.3	0.15
Pooling Mode (pooling_mode)	Mean	Sum	Mean	Mean
Feedforward Dimension (dim_feedforward)	128	256	512	1024