MLP-Mixer based Masked Autoencoders Are Effective, Explainable and Robust for Time Series Anomaly Detection

Qideng Tang
National Key Laboratory of Information Systems Engineering
National University of Defense Technology
tqd18907@nudt.edu.cn

Chaofan Dai
Yahui Wu
National University of Defense Technology
{cfdai,wuyahui}@nudt.edu.cn

Haohao Zhou*
National Key Laboratory of Information Systems Engineering
National University of Defense Technology
haohaozhou@nudt.edu.cn

ABSTRACT

Time series anomaly detection remains one of the most active research areas in data mining due to its wide range of real-world applications. In recent years, numerous deep learning-based methods have been proposed for this task. However, deep learning-based methods fail to detect subsequence anomalies with long durations, lack explainability, and are vulnerable to training set contamination. This paper addresses these issues by proposing a novel deep learning framework for effective, explainable, and robust time series anomaly detection. Our framework, **MMA**, incorporates the **M**LP-Mixer backbone with a **M**asked **A**utoencoder-based anomaly detection approach to allow for a significantly larger input window size (10 to 20 times larger than the input window sizes of current models). This larger input window enables our model to detect challenging subsequence anomalies. Meanwhile, a contrast learning module is proposed to aid in detecting subtle anomalies that fail to be identified by residual errors. Furthermore, a dynamic anomaly filtering method is introduced to mitigate the impact of subsequence anomalies on the reconstruction of surrounding normal regions to reduce false alarms. Extensive experiments on univariate and multivariate time series datasets demonstrate that our proposed framework significantly outperforms state-of-the-art methods across rigorous evaluation metrics. Additionally, MMA has a strong ability to reconstruct potential normal patterns in anomalous regions, providing high levels of explainability. Moreover, MMA demonstrates high robustness to various types of training set pollution.

PVLDB Reference Format:
Qideng Tang, Chaofan Dai, Yahui Wu, and Haohao Zhou. MLP-Mixer based Masked Autoencoders Are Effective, Explainable and Robust for Time Series Anomaly Detection. PVLDB, 18(3): 798 - 811, 2024.
doi:10.14778/3712221.3712243

PVLDB Artifact Availability:
The source code, data, and/or other artifacts have been made available at https://github.com/richard-tang199/MMA.

*Corresponding author

Proceedings of the VLDB Endowment, Vol. 18, No. 3 ISSN 2150-8097.
doi:10.14778/3712221.3712243

1 INTRODUCTION

Time series anomaly detection refers to identifying unusual patterns that significantly deviate from the majority of observations in a sequence of data collected over time. This technique is essential in various application domains, including healthcare monitoring, financial fraud detection, spacecraft telemetry, and server center operations. Due to the cost and difficulty of manual labeling work in these real-world applications, time series anomaly detection is often formulated as an unsupervised task with unlabeled training data [3]. Unsupervised anomaly detection generally presumes that the training data only contains normal samples, allowing the model to capture the normal patterns of time series. The samples that deviate from the learned normal patterns are then identified as anomalies. Classic unsupervised methods include discord discovery-based methods [36], graph-based methods [4], and density-based methods [2]. In recent years, with the rise of deep learning, many methods based on deep learning have claimed that neural networks can facilitate the learning of long-term, intricate nonlinear temporal relationships in time series, and are therefore beneficial for anomaly detection [10, 20, 41, 55, 64]. However, recent studies [25, 35, 37, 42, 43, 45, 52] have indicated that deep learning-based methods do not perform well on reputable datasets [58] when rigorous evaluation metrics are employed [14, 23, 39]. This problem has sparked skepticism about the effectiveness of deep learning methods for time series anomaly detection [18]. Our observations align with the aforementioned studies, and we find the following three limitations in current deep learning methods:

Lack of the ability to detect anomalies with prolonged durations (Limitation 1): As illustrated in Figure 1, state-of-the-art deep learning methods, including reconstruction-based methods such as TranAD [56], MAUT [41] and MTAD-GAT [66], and prediction-based methods such as CAD [50], both tend to overfit prolonged continuous anomalies. These methods utilize the residual errors (the absolute difference between the observation values and the reconstructed or predicted values) as the anomaly scores, thus failing to detect such anomalies. It is embarrassing that deep learning methods claim to model long-term temporal dependencies but fail to detect "long-term" anomalies. In our opinion, this problem stems from the inappropriate architecture design of current deep learning models, leading to a constrained input contextual window size, specifically, the input window size for reconstruction-based models and the prediction horizon for prediction-based models. According to the original papers on these methods [41, 50, 56, 66], the optimal input contextual window size is 3 for CAD, 10 for TranAD and 100 for MAUT and MTAD-GAT.

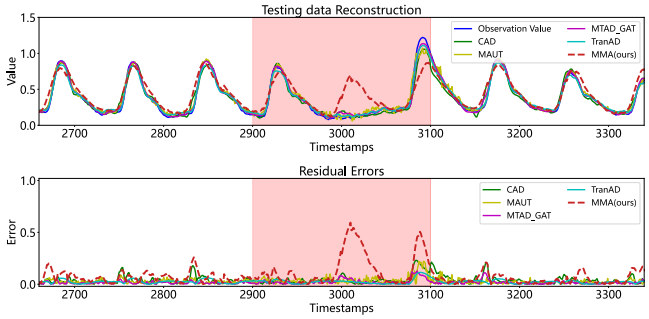


Figure 1: Detection results on the KDD21 004 dataset. The anomalous region is highlighted in light red. Our model allows for large input windows, enabling effective detection of anomalies with prolonged duration.

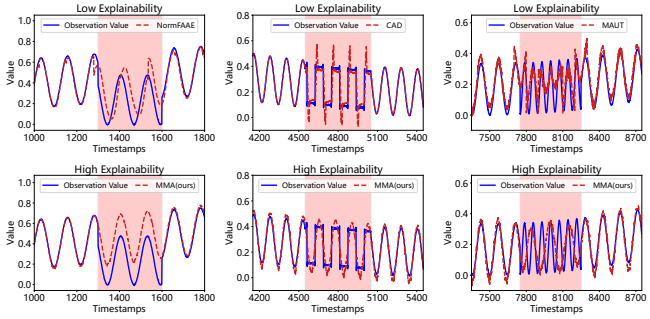


Figure 2: A comparison of reconstruction values between our high explainability model and other low explainability models on the NeurIPS-TS-Synthetic dataset.

However, as exemplified in Figure 1, when anomalies persist for relatively long durations, the input window consists only of anomalies, leading to overfitting on these anomalies due to the strong generalization capabilities of deep learning models. Increasing the window size simply does not address this issue, as larger input windows make the prediction and reconstruction more challenging [7]. This leads to high residual errors in normal regions, causing severe false alarms. In the ideal scenario, we expect a large input window size to prevent anomaly values from dominating the input while ensuring that normal values have minimal residual errors.

Lack of explainability (Limitation 2): Considering that anomaly detection models may be deployed in safety-critical domains, such as disease diagnosis and spacecraft condition monitoring, it is expected that the model will not only provide accurate detection results, but also offer tangible explanations for why a specific region is detected as an anomaly [32]. In real-world scenarios, domain experts often explain why they identify something as an anomaly by describing what it should look like if it were normal [48]. Therefore, we argue that an explainable anomaly detection model should be able to reveal the potential normal patterns in anomalous regions. As depicted in Figure 2, although both models can detect the anomalous regions through residual errors, our model with high explainability can reconstruct the possible normal patterns within the anomalous areas. In contrast, current models with low explainability can only offer noisy reconstructions.

Lack of Robustness (Limitation 3): The robustness of anomaly detection models has been a long-standing concern for researchers [20, 22, 30, 33, 46, 64]. In real applications, models are inevitably trained on datasets that may be polluted with unknown anomalies. Even if the training set contains few anomalies, unsupervised methods are likely to overfit these anomalies, often leading to performance degradation. To address this problem, many recent studies have incorporated sophisticated modules to enhance model robustness. However, these studies often lack explicit robustness evaluation [19, 20, 61, 67] or only consider adding Gaussian Noises as anomalies in the training set [1, 10, 49, 65]. Such approaches do not provide compelling evidence of model robustness, as real-world datasets are unlikely to contain only Gaussian Noises. In our experiments, various anomalies are injected into the training sets, such as Gaussian Noises, trend anomalies, seasonal anomalies, shape anomalies, uniform anomalies, and real anomalies clipped from the testing sets. Unfortunately, we find that currently available models with additional robustness designs perform poorly on these contaminated datasets.

In light of the aforementioned limitations, it appears that, as Prof. Eamonn Keogh has suggested, deep learning may not be the solution for time series anomaly detection [18]. However, our work reveals the true potential of deep learning in time series anomaly detection. We propose MMA, an MLP-Mixer based Masked Autoencoder, to achieve effective, explainable, and robust time series anomaly detection. The model employs a patch-based input scheme [38], allowing for the input of contextual windows with a length of 1024, 2048, or larger. Such a long input window prevents anomalies from dominating the input, thereby benefiting the detection of anomalies with prolonged durations. To permit a long input window size while ensuring accurate reconstruction of normal regions, a training and detection mode resembling Masked Autoencoders [13] is adopted. During training, 50% of the input patches are randomly masked, and the model is trained to reconstruct these masked patches. In the testing phase, the reconstruction errors of the masked patches are used as an indicator for identifying anomalies. Since unmasked values provide additional information, the reconstruction errors for normal regions remain small even with large input windows. Meanwhile, the MLP-Mixer backbone adopted by our model can effectively model global temporal dependencies and is more lightweight than Transformer and RNN structures. This ensures that our model remains highly efficient even with large input windows.

Furthermore, we observe some abnormal values exhibit similar amplitudes with potential normal patterns, making them difficult to detect by reconstruction errors alone. We propose a novel contrastive learning module to identify these subtle anomalies by comparing the discrepancy between their embeddings and the embeddings of reconstructed normal values. In addition, subsequence anomalies disrupt the reconstruction of neighboring normal regions, resulting in false alarms in these areas. A simple yet effective dynamic anomaly filtering method is designed to solve this problem.

We assess the performance of our model using recently proposed rigorous evaluation metrics on trustworthy datasets from multiple domains. Our model outperforms 13 state-of-the-art deep

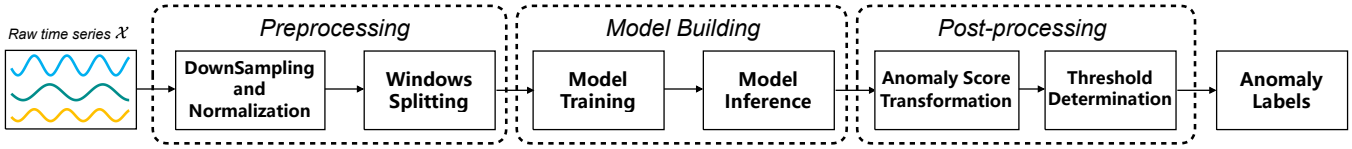


Figure 3: The Time Series Anomaly Detection Pipeline.

learning methods and 5 non-deep learning methods across both multivariate and univariate time series datasets ***(Address Limitation 1)***. Furthermore, we observe that our model exhibits a strong capability to reconstruct potential normal patterns in anomalous regions, thereby providing high levels of explainability ***(Address Limitation 2)***. In addition to offering visual evidence of explainability, we develop a post-hoc explainability analysis method to quantify the explainability of various anomaly detection models. Regarding the validation of robustness, we demonstrate that our model maintains stable performance despite various types of training set contamination without the need for any additional robustness design ***(Address Limitation 3)***.

In conclusion, we make the following contributions:

- **Exploration of new anomaly detection schemas.** To the best of our knowledge, our model is the first deep learning-based anomaly detection approach that accommodates such a large input window size. Benefitting from this large input window, our model can detect subsequence anomalies with long durations.
- **Adoption of a novel model architecture.** We choose MLP-Mixer as the backbone instead of the commonly used RNNs or Transformers, demonstrating the efficacy and efficiency of linear models. In addition, we propose a contrastive learning approach to help detect subtle anomalies and a dynamic anomaly filtering method to reduce false alarms in normal regions.
- **Rigorous explainability and robustness verification.** We design a quantitative measure for model explainability and test the robustness of models on training sets containing various types of pollution.

2 PRELIMINARIES

2.1 Problem Formulation

A time series is denoted as $\mathcal{X} = \{x_1, ..., x_T\}$, where T represents the number of observations and each observation $x_t \in \mathbb{R}^C$. C represents the number of channels in $\mathcal{X}$. When $C = 1$, $\mathcal{X}$ is a univariate time series, and when $C > 1$, $\mathcal{X}$ is a multivariate time series. In the unsupervised setting, a training time series $\mathcal{X}^{Train} \in \mathbb{R}^{C \times T_1}$ without any label indicating anomaly is given. The task is to compute an anomaly score $AS(x_t) \in \mathbb{R}$ for each observation x_t in the testing time series $\mathcal{X}^{Test} \in \mathbb{R}^{C \times T_2}$. A higher anomaly score $AS(x_t)$ indicates that the observation x_t is more likely to be an anomaly. With a predefined threshold θ_a, an observation x_t is assigned a label based on $\hat{y}_t = \mathbb{I}(AS(x_t) \geq \theta_a)$, where $\mathbb{I}(\cdot)$ is the indicator function, 1 denotes an anomaly and 0 denotes normal.

2.2 The Detection Pipeline

As depicted in Figure 3, time series anomaly detection primarily involves three processes: preprocessing, model building, and post-processing [12]. Down-sampling is generally used to reduce the number of observations in the original sequence, thereby reducing the time required for model training and testing. Window splitting divides raw, long time series into several windows of the input size allowed by the model. Anomaly score transformation combines a multichannel score such as channel-wise residual errors into a single anomaly score $AS(x_t)$ per time point. Threshold determination is to find the threshold for identifying anomalies.

Recent works indicate that both down-sampling and normalization in preprocessing [17], as well as anomaly score transformation and threshold determination in post-processing [12], have a significant impact on the final performance of models. Consequently, it is challenging to discern whether the performance improvements are attributed to the model itself or the various processing procedures. To ensure a fair evaluation, all models in our study are implemented using the following unified pipeline:

- **Downsampling and Normalization.** All models are trained and tested on datasets with the same sampling rates. All datasets are processed with the MinMax normalization method described in [56].
- **Anomaly Score Transformation.** We take the method described in [12] to aggregate multichannel anomaly scores. It first subtracts the channel-wise mean anomaly scores in the training set from the anomaly scores in the testing set to mitigate variance across channels. Subsequently, it calculates the root-mean-square of these adjusted scores across channels to obtain the aggregated anomaly scores. Finally, a moving average is applied to the aggregated anomaly scores, which amplifies the anomaly score even when multiple channels respond to an anomaly at slightly different times.
- **Threshold Determination.** To keep consistency with existing works [12, 23, 26, 27, 57], we search across all possible thresholds to find the one that yields the best result for each metric. In addition, we also consider three more practical threshold selection methods and test the performance of models under these methods.

3 METHODOLOGY

The overview of MMA is depicted in Figure 4(a), and the details are described in the following sections.

3.1 Patching and Patch Embedding

We consider each channel in the original input window $\mathcal{X} \in \mathbb{R}^{C \times T}$ as an independent univariate time series $\mathcal{X}^c \in \mathbb{R}^{1 \times T}, c = 1, 2, ..., C$. Each univariate time series $\mathcal{X}^c$ is subsequently divided into non-overlapping patches with patch length P. After that, $\mathcal{X} \in \mathbb{R}^{C \times T}$ is reshaped into $\mathcal{X}_p \in \mathbb{R}^{C \times N \times P}$, where N denotes the number of patches ($N = T/P$). We denote the i-th patch in the c-th channel as $\mathcal{X}_p^{(c,i)}$, where $\mathcal{X}_p^{(c,i)} \in \mathbb{R}^P$ and $i = 1, 2, ..., N$. The patches $\mathcal{X}_p \in$

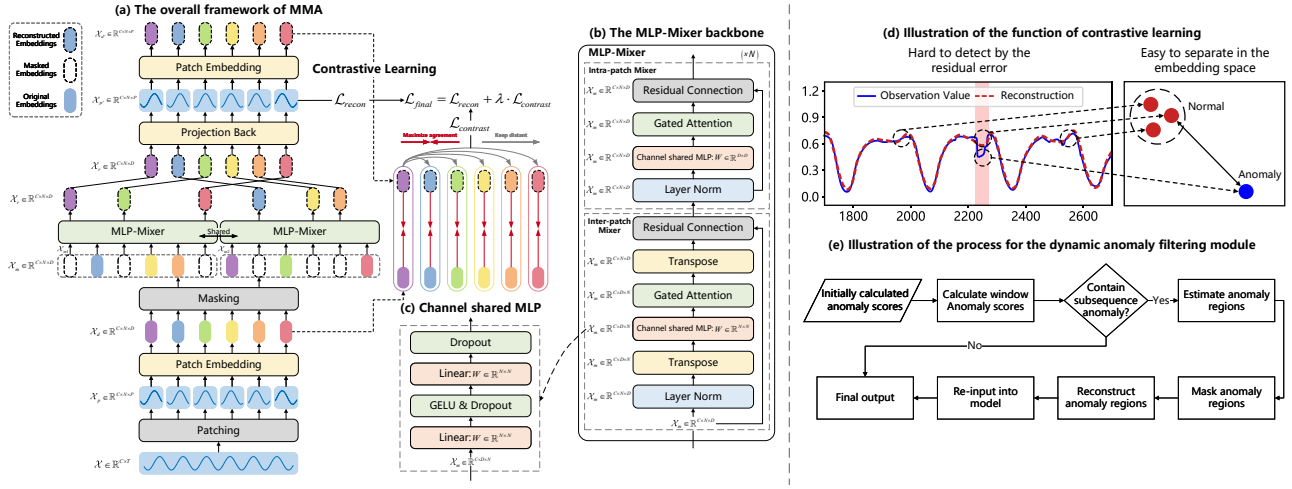


Figure 4: An introduction to the framework and submodules of MMA.

$\mathbb{R}^{C\times N\times P}$ are then mapped to embeddings $\mathcal{X}_d \in \mathbb{R}^{C\times N\times D}$ via a shared trainable linear projection layer $W_{proj} \in \mathbb{R}^{P\times D}$.

3.2 Masking

Following the default setting of the Masked Autoencoder [13], we adopt the "random sampling" masking strategy, that randomly selects patches in each channel to mask according to a uniform distribution. We use a 50% masking ratio because it only requires switching the mask once (masking previously unmasked parts in the following round) to reconstruct all patches for training and testing, making the whole process straightforward and efficient. Given that masking the latent vectors is equivalent to masking the original patches [62], the masking operation is applied directly on $\mathcal{X}_d \in \mathbb{R}^{C\times N\times D}$ to generate two masking views.

In detail, we first randomly select 50% of the patch embeddings in each channel and mask them with zeros to generate the first masking view $\mathcal{X}_{m1}$. Then, we mask the patch embeddings that were not masked in the previous step to generate the second masking view $\mathcal{X}_{m2}$. Let $\mathbb{1}$ denote the all-ones matrix, $\odot$ denotes the element-wise product and $\mathcal{M}$ represents the masking matrix, where 0 denotes positions without a mask, and 1 denotes the masked positions. The masking process can be described as follows:

$$\begin{aligned} \mathcal{X}_{m1} &= \mathcal{X}_d \odot \mathcal{M} \\ \mathcal{X}_{m2} &= \mathcal{X}_d \odot (\mathbb{1} - \mathcal{M}) \end{aligned} \tag{1}$$

3.3 MLP-Mixer backbone

The two masking views $\mathcal{X}_{m1}$ and $\mathcal{X}_{m2}$ are then input to the MLP-Mixer backbone shown in Figure 4(b) to recover the masked embeddings. The MLP-Mixer backbone is modified from [11, 29]. We retain the inter-patch mixer and intra-patch mixer modules while removing the inter-channel mixer module to avoid over-smoothing across channels. The **Inter-patch Mixer** block employs a channel shared MLP with weights $W \in \mathbb{R}^{N\times N}$ to capture the correlation between different patches, thereby facilitating the learning of global temporal dependencies among patches. The **Intra-patch Mixer** block takes a channel shared MLP with weights $W \in \mathbb{R}^{D\times D}$ to mix the hidden features of time steps within a patch, thereby enabling the learning of local temporal dependencies in a patch.

The structure of each channel-shared MLP is depicted in Figure 4(c), which consists of two fully connected layers, a Gelu nonlinear layer, and two Dropout layers. A Gated Attention layer is added after each MLP component to filter out noise in the time series. The Gated Attention layer upscales dominant features and downscales unimportant features based on their feature values. For instance, the computation process of the Gated Attention in the Intra-patch Mixer block is formulated as follows:

$$\begin{aligned} \mathcal{W} &= \text{softmax}\left(\mathrm{W}(\mathcal{X}_m)\right), \mathrm{W} \in \mathbb{R}^{D\times D} \\ \mathcal{X}'_m &= \mathcal{W} \odot \mathcal{X}_m, \mathcal{W} \in \mathbb{R}^{C\times N\times D} \end{aligned} \tag{2}$$

where $\mathcal{W}$ represents the learned attention weights, and the output is the element-wise product of the attention weights and the original embeddings.

3.4 Contrastive Learning

As shown in Figure 4(d), some anomalies in the time series have amplitudes similar to the normal patterns but exhibit distinct shapes. Therefore, it is difficult to detect them solely based on residual errors. To address this issue, a contrastive learning module is designed to cluster the embeddings of time series patches with similar shapes in the latent vector space. Subsequently, these subtle anomalies can be detected by comparing the embeddings of the observed values with the embeddings of the reconstructed values. However, constructing proper positive and negative samples is a challenging task for time series contrastive learning [62].

In our work, as illustrated in Figure 4(a), we consider the embeddings derived directly from the raw patches and the embeddings obtained from the reconstructed patches as positive pairs, while the embeddings derived from different patches are treated as negative samples. It is worth noting that the two Patch Embedding modules in Figure 4(a) are shared. Let $\mathcal{X}_d^{(c,i)}$ denotes the embedding directly obtained from patch $\mathcal{X}_p^{(c,i)}$, and $\mathcal{X}_{d'}^{(c,i)}$ denotes the

embedding derived from the reconstructed patch $\mathcal{X}_{p'}^{(c,i)}$. The contrastive loss for patch $\mathcal{X}_{p}^{(c,i)}$ can be written as:

$$\mathcal{L}_{contrast}^{(c,i)} = -\log \frac{\exp(\mathcal{X}_{d}^{(c,i)} \circ \mathcal{X}_{d'}^{(c,i)})}{\sum_{i' \in I} \mathbb{I}_{[i' \neq i]} \exp(\mathcal{X}_{d}^{(c,i)} \circ \mathcal{X}_{d'}^{(c,i')})} \tag{3}$$

where the cosine similarity $\circ$ is used as the distance metric, $\mathbb{I}$ is the indicator function, and I represents the set of patches within the channel c. Finally, the overall contrastive loss for all patches in all channels can be written as:

$$\mathcal{L}_{contrast} = \frac{1}{N * C} \sum_{i=1}^{N} \sum_{c=1}^{C} \left(\mathcal{L}_{contrast}^{(c,i)} \right) \tag{4}$$

Our proposed contrastive learning module avoids using inappropriate positive samples constructed through traditional data augmentation methods [21, 52]. It enables the patch embeddings to be aware of the shape information, thereby serving as a criterion for detecting anomalies.

3.5 Training and Anomaly Scoring

We denote the embeddings recovered from the masking views $\mathcal{X}_{m1}$ and $\mathcal{X}_{m2}$ as $\mathcal{X}_{r1}$ and $\mathcal{X}_{r2}$, respectively. The embedding $\mathcal{X}_{r1}$ and $\mathcal{X}_{r2}$ are merged in chronological order to form the overall reconstructed embedding $\mathcal{X}_r$. $\mathcal{X}_r$ is subsequently mapped back to the time series patches $\mathcal{X}_{p'}$, via a shared trainable linear projection layer with weights $W \in \mathbb{R}^{D \times P}$. The reconstruction loss is the Mean Squared Error (MSE) between the original patches $\mathcal{X}_p$ and the reconstructed patches $\mathcal{X}_{p'}$:

$$\mathcal{L}_{recon} = \frac{1}{T * C} \left\| \mathcal{X}_p - \mathcal{X}_{p'} \right\|_2^2 \tag{5}$$

The overall training loss is the weighted sum of the reconstruction loss and the contrastive loss:

$$\mathcal{L}_{final} = \mathcal{L}_{recon} + \lambda * \mathcal{L}_{contrast} \tag{6}$$

During the inference stage, the anomaly score is composed of two components: (1) the residual error between the original patch and the reconstructed patch; and (2) the cosine distance between the original embedding and the embedding derived from the reconstructed patch. The anomaly score $AS\left(\mathcal{X}_p^{(c,i)}\right) \in \mathbb{R}^P$ for the patch $\mathcal{X}_p^{(c,i)}$ is written as follows:

$$AS\left(\mathcal{X}_p^{(c,i)}\right) = \left| \mathcal{X}_p^{(c,i)} - \mathcal{X}_{p'}^{(c,i)} \right| + \left(1 - \mathcal{X}_d^{(c,i)} \circ \mathcal{X}_{d'}^{(c,i)}\right) \tag{7}$$

It is worth noting that the residual error $\left| \mathcal{X}_p^{(c,i)} - \mathcal{X}_{p'}^{(c,i)} \right| \in \mathbb{R}^P$, while $\left(\mathcal{X}_d^{(c,i)} \circ \mathcal{X}_{d'}^{(c,i)} \right) \in \mathbb{R}$. To ensure dimensional consistency, we consider the points within a patch to share the same cosine distance and repeat the cosine distance value for P times, where P is the patch length. In addition, we propagate the input window forward 5 times and take the average of the anomaly scores to reduce the uncertainty caused by random masking. This can also be regarded as an "ensemble" of detection results under varying masking scenarios. Finally, the anomaly scores of each patch are concatenated along the time dimension to obtain the anomaly scores for all points within a channel.

3.6 Dynamic Anomaly Filtering

The subsequence anomaly within a window provides wrong information during the reconstruction of their surrounding normal regions, resulting in high reconstruction errors in these areas. We add a dynamic anomaly filtering module during the inference stage to address this issue. This module first estimates the position of the subsequence anomaly within a window based on the initially calculated anomaly scores, and then replaces the anomalous regions with normal values restored from the MLP-Mixer backbone. The processed window is subsequently re-input into the model for a second reconstruction. We calculate the final anomaly scores based on the second reconstruction values and the initial input values using Equation 7. During this second reconstruction, most of the anomalous regions within the input window have been previously replaced by the reconstructed normal values, which greatly diminishes the impact of subsequence anomalies on the reconstruction of surrounding normal areas, thereby reducing false alarms in these regions. The overall workflow of this module is illustrated in Figure 4(e) and the details are described as follows:

For a given univariate window $\mathcal{X}^c$, we first calculate the anomaly score $AS\left(\mathcal{X}^c\right)$ for this window by summing the anomaly scores of all points within the window. Moreover, in this step of calculating anomaly scores, we only adopt the residual errors to reduce computational load. If a window contains a subsequence anomaly, its anomaly score will be significantly higher than that of other windows. Therefore, we employ the 3σ principle to determine whether the window contains subsequence anomalies or not. The steps are formulated as follows:

$$\begin{aligned} AS\left(\mathcal{X}^c\right) &= \sum_{i=1}^{N} \sum AS\left(\mathcal{X}_p^{(c,i)}\right) \\ y &= \mathbb{I}\left(AS\left(\mathcal{X}^c\right) \geq \mu_{train} + 3 * \sigma_{train}\right) \end{aligned} \tag{8}$$

where $\sum AS\left(\mathcal{X}_p^{(c,i)}\right) \in \mathbb{R}$ represents the summation of anomaly scores for all points within a patch. μ_{train} and σ_{train} are the mean and standard deviation of the anomaly scores for windows in the training set. If $y = 1$, we assume the window contains subsequence anomalies and continue to apply the 3σ principle for this window to search for patches that may contain anomalies:

$$I_a = \left\{ \mathcal{X}_p^{(c,i)} \mid \sum AS\left(\mathcal{X}_p^{(c,i)}\right) \geq \mu'_{train} + 3 * \sigma'_{train}, i = 1, 2, ..., N \right\} \tag{9}$$

where I_a represents the set of patches in window $\mathcal{X}^c$ that potentially contain anomalies. μ'_{train} and σ'_{train} are the mean and standard deviation of the summation anomaly scores for patches in the training set. Since anomalies are rare events, we assume that a window contains only one subsequence anomaly. Therefore, we consider the median of the positions for the patches in I_a as the center of the subsequence anomaly, and the sum length of the patches in I_a as the length of the subsequence anomaly. Let I_p denotes the set of positions of patches from I_a. The estimated anomalous region p_a can be expressed as follows:

$$\begin{aligned} p_c &= \text{median}\left(I_p\right) \\ p_a &= \left[p_c - \frac{P}{2} * |I_a|, p_c + \frac{P}{2} * |I_a| \right] \end{aligned} \tag{10}$$

where p_c denotes the estimated center of the anomalous region, $|I_a|$ represents the number of patches in the set I_a, and $P * |I_a|$ represents the estimated anomaly length. After obtaining the anomalous region p_a, we mask this region and utilize the MLP-Mixer backbone to recover the masked parts. After that, most of the anomalous regions within the input window are replaced by the reconstructed normal values. Finally, we re-input the processed window back into the model for a second reconstruction and recalculate the anomaly score for each point in this window.

4 EXPERIMENTS AND ANALYSIS

In this section, we first introduce the benchmark datasets, baseline methods, and the implementation details of models. Then, we conduct experiments to answer the following research questions:

- **RQ1. Effectiveness.** How does MMA perform on time series anomaly detection datasets compared to other state-of-the-art approaches?
- **RQ2. Explainability.** How accurately does MMA recover the potential normal patterns for anomalous regions compared to other models?
- **RQ3. Robustness.** Can MMA maintain stable performance in the presence of various kinds of training set contamination?
- **RQ4. Ablation.** How much does each component in MMA contribute to the overall performance?
- **RQ5. Visualization.** Can MMA provide detection results that align with human institutions?
- **RQ6. Insights.** What insights can be gained from our method?

4.1 Datasets and Baselines

Benchmark Datasets. The quality of benchmark datasets is a major concern in the field of time series anomaly detection. Through a comprehensive review of recent benchmark evaluation papers [16, 40, 44, 53, 57, 58], and a meticulous examination of the visualized datasets, we choose a set of relatively high-quality datasets from diverse domains as benchmarks. These datasets include the univariate time series anomaly detection dataset: the KDD21 dataset [58], and the multivariate time series anomaly detection datasets: ASD [31] and NeurIPS-TS-Synthetic (Synthetic) [25]. Moreover, we use a real-world dataset collected from a satellite. The statistics of the datasets are summarized in Table 1.

The KDD21 dataset comprises entities from various domains, including healthcare, sports, industry, robotics, etc. The ASD dataset is collected from Internet server machines. The NeurIPS-TS-Synthetic dataset utilizes the sinusoidal wave as the base shapelet and injects multiple predefined anomalies into it. These datasets are less affected by issues such as unrealistic anomaly density, position bias, distribution shift, and mislabeled ground truth, as mentioned in [57, 58]. Therefore, they are more suitable for performance evaluation.

Baseline Methods. We compare the proposed MMA with 18 state-of-the-art baselines including 13 deep learning-based models and 5 non-deep learning models.

Deep learning-based models. We consider deep learning models with various detection strategies: (1) The reconstruction-based methods (e.g., NSPR [24], NormFAAE [61], MAUT [41], TranAD [56], FGANomaly [10], USAD [1] and MTAD-GAT [66]) build models to reconstruct the time series from hidden vectors and identify anomalies based on the reconstruction errors; (2) The prediction-based models (e.g., CAD [50] and GDN [9]) learn to predict the future values of the time series and detect anomalies based on the prediction errors; (3) The imputation-based methods (e.g., DiffAD [59] and ImDiffusion [7]) learn to impute the missing values based on the observed values and find anomalies using the imputation errors; (4) The density-based methods (e.g., MTGFLOW [69]) evaluate the density of the time series and treat anomalies as regions with low density. (5) The embedding-based methods (e.g., PatchAD [68]) distinguish normal and abnormal data based on their latent embeddings. In addition, NormFAAE, FGANomaly, USAD, MTAD-GAT, and MTGFLOW utilize RNNs as their backbone, while NSPR and TranAD employ Transformers as the backbone. DiffAD and ImDiffusion are based on diffusion models. CAD uses CNN to model temporal dependencies, and GDN leverages graph neural networks for learning temporal information. PatchAD also takes the MLP-Mixer as its backbone.

Table 1: Datasets used in this study before preprocessing.

Datasets	Entities	Dims	Train	Test	Anomaly	Domain
KDD21	250	1	2238349	6143541	0.6%	Multiple
ASD	12	19	102331	51840	4.61%	Servers
Satellite	3	9	11862	3793	7.22%	Telemetry
Synthetic	1	5	10000	10000	13.06%	Simulated

Non-deep learning models. Matrix Profile [60] identify subsequences with large distances to their nearest neighbors as anomalies. DAMP [35] is an improved version of the original matrix profile, designed for online detection and processing ultra-fast arriving time streams. SAND [5] is a cluster-based model that detects anomalies based on the distance to a model representing normal behaviors. Series2Graph [4] aims to detect subsequence anomalies based on a graph representation of a low-dimensionality embedding of raw time series. k-Means [54] is based on subsequence clustering where samples that are far from the cluster centers are considered as anomalies.

4.2 Implementation Details.

Hyperparameters Settings. In all experiments, we set the MLP-Mixer layers to 3, the hidden dimension D to 64, and the weight λ to 0.005. For the ASD, Satellite, and NeurIPS-TS-Synthetic datasets, the input window size T is set to 1024, and the patch length P is set to 16. Since the KDD21 dataset contains entities with different sampling frequencies, we use different input window sizes for each entity. We first apply the Fast Fourier Transform to identify the primary period of each entity. Then, the input window size is set to 4 times the primary period and the patch length is set to the period divided by 8. We train our model for 200 epochs on the KDD21 dataset and 100 epochs on other datasets.

Baseline Settings. We implement all the deep learning-based models based on their official open-source codes and integrate them into our unified pipeline. All non-deep learning models except k-Means are implemented based on the TSB-UAD library [40]. k-Means is implemented based on the TimeEval library [44]. The hyperparameters of the baseline models are set according to the information provided in their original papers.

Table 2: Results on multivariate datasets. All results are in %, the best ones are in Bold, and the second ones are underlined.

Method	ASD						Synthetic						Satellite					
	AUC-PR	R-AUC-PR	VUS-PR	F1-Raw	F1-(PA%K)	F1-Aff	AUC-PR	R-AUC-PR	VUS-PR	F1-Raw	F1-(PA%K)	F1-Aff	AUC-PR	R-AUC-PR	VUS-PR	F1-Raw	F1-(PA%K)	F1-Aff
NSPR	42.83	36.65	36.20	50.68	54.90	78.08	17.93	29.17	29.17	31.84	47.28	69.52	68.68	68.12	67.85	73.26	85.75	89.77
CAD	44.26	46.16	45.37	45.71	57.91	85.81	66.21	74.73	74.59	69.55	83.77	92.23	73.23	72.31	71.95	74.77	90.25	98.52
DiffAD	6.17	10.07	10.01	11.36	20.83	71.69	15.15	26.59	26.81	24.21	43.13	67.93	17.94	27.77	26.11	27.72	48.50	80.92
ImDiffusion	19.41	22.10	21.74	19.28	29.23	N/A	31.53	44.29	43.60	23.26	23.91	N/A	13.66	19.26	19.39	22.12	13.48	N/A
NormFAAE	28.58	27.99	27.80	36.20	47.27	81.91	39.48	45.21	44.40	37.71	50.75	77.96	60.00	59.01	58.44	62.15	80.39	97.50
MTGFLOW	5.79	8.65	8.49	12.38	17.45	69.55	11.08	21.02	20.93	24.18	34.92	70.41	20.92	32.81	32.82	28.68	42.67	80.47
MAUT	39.52	40.00	39.58	45.33	56.63	83.21	67.63	67.90	66.15	63.28	78.77	87.40	55.40	57.65	57.05	64.59	81.93	89.03
TranAD	36.42	38.78	38.63	41.95	55.25	83.46	13.07	23.81	22.99	23.12	36.09	69.69	81.32	74.65	75.40	78.30	91.86	**99.55**
FGANomaly	35.98	35.41	35.36	40.29	52.44	82.34	18.30	27.80	27.90	28.77	44.94	69.09	86.11	74.50	76.42	81.18	94.24	98.62
GDN	31.07	28.24	28.31	38.15	51.61	79.92	15.03	22.18	23.07	23.56	39.38	74.99	62.97	61.02	61.20	70.30	86.37	92.17
USAD	33.21	27.47	27.17	36.91	45.96	77.58	14.42	20.94	21.16	23.11	33.79	69.78	69.26	67.24	66.24	74.32	85.47	88.94
MTAD-GAT	35.99	36.63	36.43	40.81	54.74	81.89	76.56	76.62	76.76	77.39	88.53	93.65	54.16	54.54	53.92	64.98	81.04	86.53
PatchAD	15.21	12.48	12.49	21.93	31.08	71.98	14.14	22.58	23.45	23.22	40.36	67.92	49.55	47.86	48.07	53.08	73.74	87.17
k-Means	54.70	**59.64**	57.57	54.39	63.67	87.06	27.31	34.48	33.36	31.67	48.96	69.64	82.66	79.27	79.17	78.07	91.56	99.23
MMA(ours)	**58.08**	58.29	**57.67**	**58.23**	**69.59**	**89.71**	**88.67**	**90.96**	**90.62**	**88.93**	**95.82**	**93.72**	**92.46**	**87.51**	**87.72**	**87.66**	**96.89**	99.50

Table 3: Average results on the KDD21 dataset.

Method	KDD21					
	AUC-PR	R-AUC-PR	VUS-PR	F1-Raw	F1-(PA%K)	F1-Aff
NSPR	8.37	8.28	8.27	12.07	16.75	72.41
CAD	23.90	23.25	23.03	30.03	33.88	79.37
NormFAAE	3.12	3.76	3.66	6.39	11.59	70.26
MAUT	23.84	23.94	23.65	30.20	35.45	80.34
TranAD	7.78	7.94	7.89	11.23	12.81	72.85
FGANomaly	8.65	8.15	8.10	12.79	13.34	73.28
USAD	8.33	8.37	8.34	12.16	15.21	73.96
MTAD-GAT	26.00	26.18	25.66	38.04	36.99	78.05
DAMP	18.99	25.09	24.18	29.31	29.95	84.88
SAND	34.72	34.23	33.78	39.43	47.54	84.07
Series2Graph	4.36	8.18	7.87	8.70	11.94	78.42
Matrix Profile	18.50	25.37	24.13	28.00	31.49	80.58
k-Means	34.33	33.49	33.20	37.97	44.59	83.70
MMA(ours)	**39.24**	**37.97**	**37.38**	**44.47**	**52.36**	**87.27**

4.3 RQ1. Effectiveness

Evaluation Metrics. Since there is currently no universally accepted fair and rigorous evaluation metric for time series anomaly detection, we adopt some commonly used metrics as well as recently proposed metrics specifically designed for time series anomaly detection. In light of the severe flaws in the widely used point adjustment strategy [14, 15, 23, 57], all our results are reported **without point adjustment.** Due to the significant class distribution skew in the anomaly detection datasets [8, 28, 39], we opt to use AUC-PR (Area Under the Precision-Recall Curve) instead of AUC-ROC (Area Under the Receiver Operating Characteristic Curve). Here is a brief introduction to the evaluation metrics:

The F1-Raw and AUC-PR are the most widely used evaluation metrics that calculate the point-wise F1 value and the area under the precision-recall curve. The R-AUC-PR and VUS-PR metrics [39] are recently proposed to address the inconsistent labeling problem in the evaluation of time series anomaly detection. They add a buffer at the boundary of anomalies, thereby giving some credit to the high anomaly score in the vicinity of the anomaly boundary. F1-(PA%K) [23] optimizes original point-wise evaluation by considering the balance between conventional F1-Raw measurements and the ill-posed point adjustment strategy. F1-Affiliation [14] tackles the problems of unawareness of temporal adjacency and unawareness of anomaly durations in current evaluation methods by identifying the local affiliation of predicted anomalies to their closest ground truths.

Results Analysis. The comparison results with SOTA models are presented in Table 2 and Table 3. As DAMP, SAND, Series2Graph, and Matrix Profile only accept univariate time series as input, we do not evaluate them on the multivariate time series datasets. Due to the large size of the KDD21 dataset, we only consider the top 9 performing methods in Table 2 for evaluation on the KDD21 dataset. Since most of the baselines [1, 10, 24, 41, 50, 61, 69] take the threshold that yields the best results on the testing set for each metric, to keep consistency with existing works, the F1-Raw, F1-(PA%K) and F1-Aff results in Table 2 and Table 3 are also derived using this threshold selection method. Summary from the tables, we have the following two observations:

First, several studies indicate that an individual anomaly detection model typically performs well only on a specific domain of datasets [44, 53]. This declaration aligns with our experimental results, such as k-Means demonstrating excellent performance on the ASD dataset, MTAD-GAT performing well on the NeurIPS-TS-Synthetic dataset, and FGANomaly excelling on the Satellite dataset. However, our model achieves the best results across all datasets, thereby validating the efficacy of our proposed methodology in handling time series with diverse patterns. Specifically, the KDD21 dataset comprises subsequence anomalies that are challenging to detect. Existing deep learning models generally perform poorly on this dataset [35, 40, 42]. Nevertheless, our work is the first framework that relies solely on deep learning and achieves better performance than state-of-the-art non-deep learning methods on the KDD21 dataset.

Second, some recently proposed deep learning models with intricate architectures, such as DiffAD, ImDiffusion, MTGFLOW and PatchAD perform poorly on reliable datasets when rigorous evaluation methods are employed. This causes the "Creating the Illusion of Progress" problem as suggested in [18, 58]. Moreover, we find that the F1-Affiliation metric tends to overestimate the performance of models. Even when models perform poorly on other metrics, F1-Affiliation consistently assigns high scores to them. Future researchers should be cautious in using the F1-Affiliation metric.

Threshold Analysis. In real-world scenarios, testing sets are not available beforehand for threshold determination. Therefore, we consider several more practical threshold selection methods and evaluate the performance of models using these methods. Following [16, 59, 64], we determine thresholds based on anomaly scores calculated from the training set (thresholds for KDD21 are

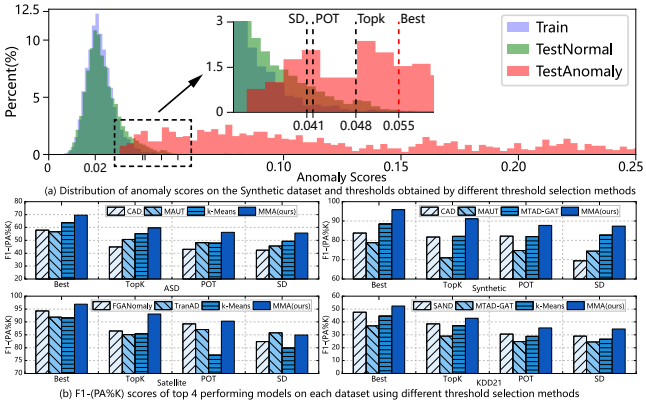


Figure 5: Threshold Analysis.

calculated from the testing set due to distribution shifts). The details of these methods are described as follows: (1) Standard Deviation (SD): The threshold is set to the mean value plus 3 times the standard deviation of the anomaly scores. (2) Peak-over-Threshold (POT) [51]: This method employs "extreme value theory" to fit the tail portion of the anomaly scores with a Generalized Pareto Distribution (GPD), and then sets the threshold based on a probability of less than 1% for anomalies to occur in this distribution. (3) TopK: Assuming we have prior knowledge of the estimated proportion of anomalies as k (%), the threshold is set as the anomaly scores ranking in k in the testing set. We use the exact proportion of anomalies in the testing set as k for an ideal situation.

As shown in Figure 5(b), MMA outperforms state-of-the-art approaches in almost all cases. This demonstrates that MMA is robust to threshold selection methods. Specifically, for MMA, TopK achieves results closest to the best threshold (the threshold that yields the highest F1-(PA%K) score). However, TopK requires preliminary information and anomaly scores for the full testing set before thresholding, making it only applicable to non-streaming data. POT performs slightly better than SD, but as shown in Figure 5(a), the thresholds chosen by POT are very close to those selected by SD in most cases. POT requires additional computation to fit the GPD distribution, whereas SD is highly efficient and delivers comparable performance. Therefore, we recommend using the SD threshold selection method in most cases. Additionally, from Figure 5(a), we find that a slightly larger threshold generally produces better results. This is because F1-(PA%K) is an event-wise metric [12, 23], that considers several alarms sufficient for a contiguous anomaly event. Raising the threshold does not miss anomalies (points with large anomaly scores still alarm this event), but significantly reduces false alarms.

4.4 RQ2. Explainability

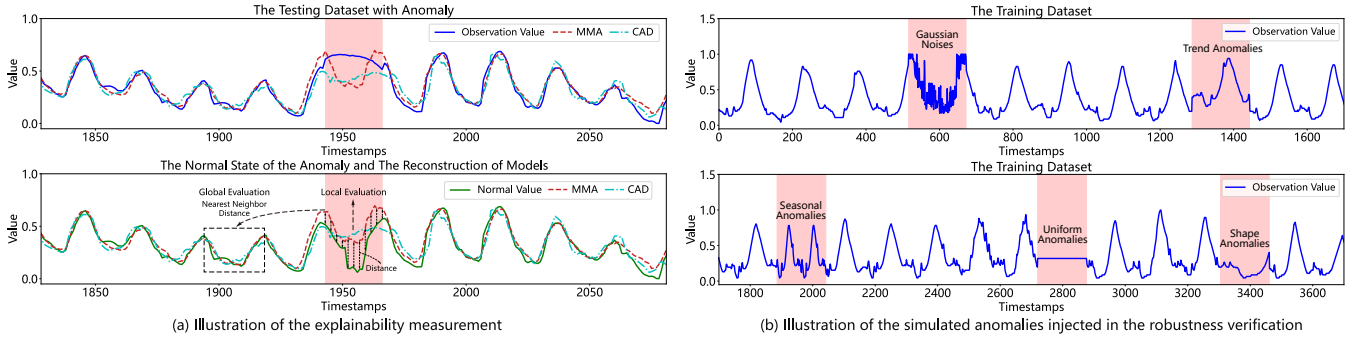
Experiment Setting. As shown in Figure 6(a), we choose some normal regions in the original time series and replace them with subsequence anomalies. Subsequently, we apply different models to reconstruct the chosen regions. Supposing the set of raw values (normal values) of the chosen regions is denoted as $S_{raw} = \{s^1_{raw}, s^2_{raw}, ..., s^n_{raw}\}$, where s^i_{raw} represents the normal value of a selected region and n is the number of chosen regions. The set of reconstructed values of the model in these regions is denoted as $S_{recon} = \{s^1_{recon}, s^2_{recon}, ..., s^n_{recon}\}$. As depicted in Figure 6(a), we consider the following two methods for measuring the explainability of models. **(1) Local Evaluation:** we calculate the mean distance between the raw values and the reconstructed values in the chosen regions: $E_{local} = \frac{1}{n}\sum_{i=1}^{n}\left(dis\left(s^i_{raw}, s^i_{recon}\right)\right)$, where $dis\,(\,,\,)$ represents the distance function. A smaller E_{local} indicates a stronger ability of the model to restore the normal states for the anomalous regions, thereby providing higher explainability. **(2) Global Evaluation:** we calculate the mean distance between the reconstructed subsequences and their nearest neighbors in the normal parts of the time series: $E_{global} = \frac{1}{n}\sum_{i=1}^{n}\left(dis\left(s^i_{recon}, s^i_{NN}\right)\right)$, where s^i_{NN} denotes the nearest neighbor subsequence of s^i_{recon} (with the same length). A small E_{global} means that the model can reconstruct a subsequence that is close to existing normal data. We select the Euclidean distance (ED) and Dynamic Time Warping (DTW) distance as the distance functions. For multivariate time series, we sum up the evaluation scores of each channel as the final score. In addition, we also conduct the aforementioned evaluations on other unmodified normal regions of identical lengths for comparison.

Results Analysis. The explainability evaluation results are presented in Figure 7. This figure clearly shows that regardless of the evaluation method used, our model consistently achieves lower evaluation scores on anomalous regions compared to other models. Furthermore, the evaluation scores of other models exhibit a substantial difference between normal and anomalous regions, whereas the discrepancy in our model's scores between the two kinds of regions remains minimal. This indicates that the reconstructions of other models are easily disrupted by anomalies, whereas our model is capable of accurately reconstructing the normal data when anomalies are given as input. By comparing the observed anomalous values with the reconstructed normal patterns from our model, operators can confirm why a certain region is detected as an anomaly.

4.5 RQ3. Robustness

Experiment Setting. As shown in Figure 6(b), the training set is polluted with various kinds of simulated anomalies, including Gaussian Noise, trend anomalies, seasonal anomalies, uniform anomalies, and shape anomalies. The ratio of injected simulated anomalies is gradually increased from 2% to 15% to observe the performance of the models under different levels of contamination. Furthermore, we inject real anomalies clipped from the testing set into the training set and repeat these anomalies 1 to 4 times in the training set to simulate varying levels of real anomaly contamination scenarios. This is considered a challenging test for the robustness of the models, as it explicitly forces them to overfit the anomalies in the testing set.

For the robustness tests conducted on the multivariate time series datasets, three models with additional robustness designs: NormFAAE, FGANomaly, and USAD, along with 2 top performing baselines on each dataset are chosen for comparison with our proposed model. For the univariate KDD21 dataset, since NormFAAE, FGANomaly, and USAD fail to detect anomalies within it, only the



(a) Illustration of the explainability measurement

(b) Illustration of the simulated anomalies injected in the robustness verification

Figure 6: Illustrations of the experimental settings for explainability and robustness verification. The data in (a) and (b) are derived from KDD21 006 and Satellite 1, respectively.

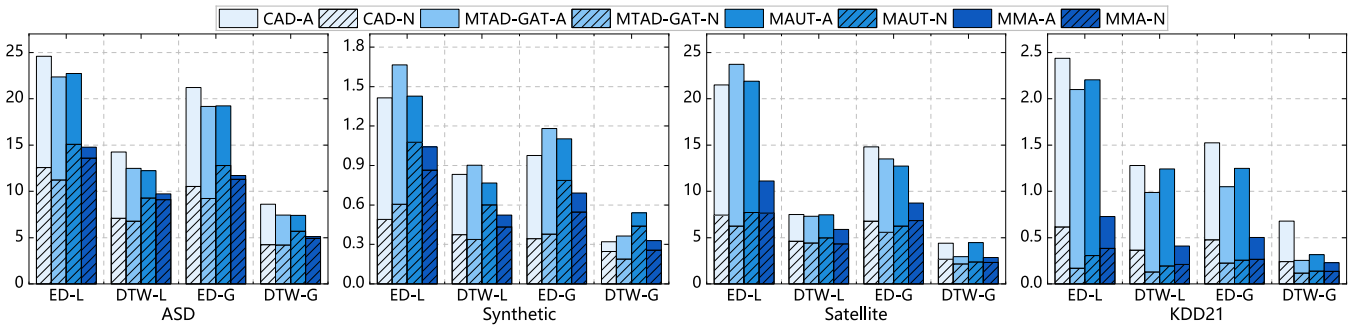


Figure 7: The explainability evaluation results on top 4 performing deep learning methods. ED-L (DTW-L) and ED-G (DTW-G) represent local and global evaluations using the ED (DTW) distance, respectively. "Model"-A (e.g., MMA-A) denotes the evaluation scores on anomalous regions, while "Model"-N (e.g., MMA-N) refers to the evaluation scores on normal regions.

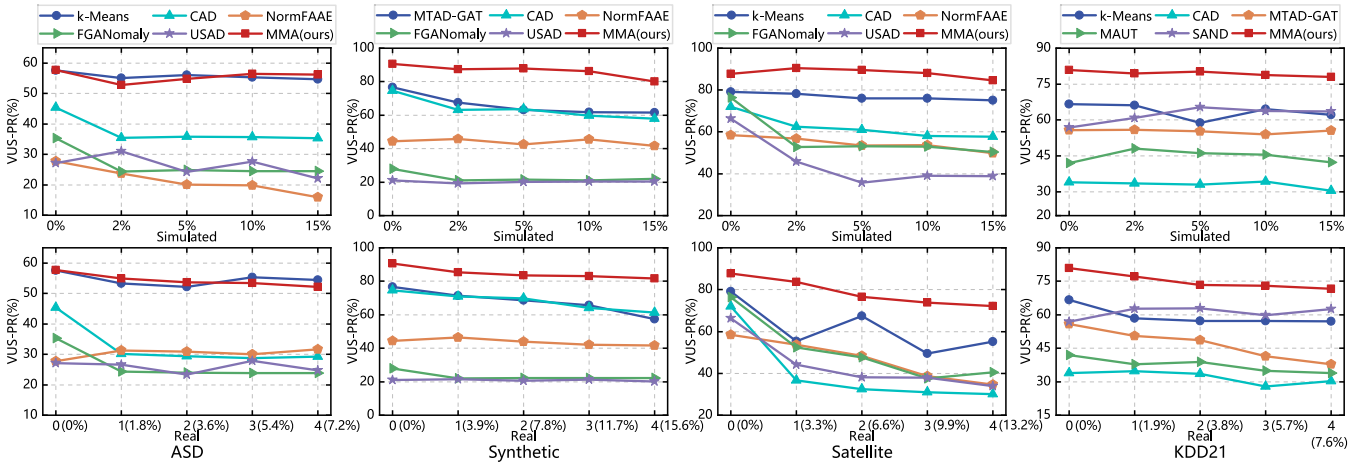


Figure 8: The robustness evaluation results. Due to the extensive size of the KDD21 dataset, we only include entities 006, 025, 048, 141, 145, 160, and 173 for testing. k% represents the ratio of the sum of anomaly lengths to the training set length.

top 6 performing models on this dataset are included for comparison.

Results Analysis. Figure 8 shows that most of the baselines experience a sharp decline in performance when the training set is contaminated with anomalies. Furthermore, the most significant drop in model performance occurs when real anomalies are injected. Despite incorporating specialized robustness designs, NormFAAE, FGANomaly, and USAD still face severe performance

losses when trained on contaminated datasets. In comparison, our proposed model is robust to anomalies in the training set, whose performance loss is less than 10% in most contamination scenarios. However, we also observe that our model experiences relatively larger performance losses on the Satellite and KDD21 datasets under high levels of real anomaly contamination. This is because the two datasets contain subtle subsequence anomalies with long durations, and it is relatively easy for our model to overfit such anomalies when they appear multiple times in the training set.

4.6 RQ4. Ablation

To study the effectiveness of each proposed component and justify the design choices, we evaluate the performance of three distinct ablated variants of our model across all datasets. **(1) MMA with different backbones.** *MMA_GRU:* We replace the MLP-Mixer backbone in MMA with Gate Recurrent Units (GRU) to model temporal dependencies between patches, similar to SegRNN [34]. *MMA_Transformer:* We use a Transformer to model the temporal dependencies between patches, similar to PatchTST [38]. **(2) MMA with different masking strategy.** *MMA_Grating:* Following ImDiffusion [7], We mask the patches at equal intervals along the time dimension, resulting in a staggered appearance of masked and unmasked patches. **(3) MMA with disabled submodules.** *MMA w/o CL:* We remove the Contrastive Learning (CL) module in MMA and the anomaly score is calculated solely based on the residual error. *MMA w/o DAF:* We remove the Dynamic Anomaly Filtering (DAF) module, disregarding the impact of anomalies on the reconstruction of neighboring normal regions. *MMA w/o CL&DAF:* We remove both the Contrastive Learning module and the Dynamic Anomaly Filtering module. The performance of MMA's variants is summarized in Table 4.

Compared to variants using other backbones, the original model with the MLP-Mixer backbone performs the best. This superiority is attributed to the MLP-Mixer's inherent sensitivity to sequence order (swapping two inputs yields different outputs) and its ability to model global temporal dependencies. It effectively leverages global information to reconstruct masked patches, thereby preventing overfitting in anomalous regions. Conversely, the gating mechanism in the GRU makes *MMA_GRU* consistently focus on the data surrounding the masked regions, while forgetting information that is farther away. As a result, when the masked areas are surrounded by anomalies, *MMA_GRU* tends to overfit these anomalies rather than reconstructing the possible normal values. Although *MMA_Transformer* can also utilize global information, the self-attention mechanism in Transformer is permutation-invariant and "anti-order" [63], necessitating the learning of additional position embeddings to capture temporal dependencies in time series [47]. When the time series contain diverse patterns but with limited training data (e.g., entities 013, 072 in KDD21), the *MMA_Transformer* struggles to learn an appropriate position embedding, thus failing to reconstruct the masked patches.

Considering the masking strategy, *MMA_Grating* results in a significant performance drop across all datasets. The grating masking reduces the reconstruction difficulty and makes the model simply restore the masked patches through extrapolation from the visible neighboring patches. This is highly undesirable for anomaly detection, as it makes the reconstruction of masked patches in anomalous regions susceptible to nearby unmasked anomalies, leading to overfitting on these anomalies.

Table 4: Performance of MMA and its variants.

Method	ASD		Synthetic		Satellite		KDD21	
	VUS-PR	F1-(PA%K)	VUS-PR	F1-(PA%K)	VUS-PR	F1-(PA%K)	VUS-PR	F1-(PA%K)
MMA	**57.67**	**69.59**	**90.62**	**95.82**	**87.72**	96.89	**37.38**	**52.36**
MMA_GRU	41.38	57.57	84.63	94.25	67.10	87.12	34.07	46.39
MMA_Transformer	34.65	53.99	67.03	80.47	65.25	87.53	24.11	32.28
MMA_Grating	43.07	54.51	79.62	88.93	82.26	88.01	30.47	45.10
MMA w/o CL	42.38	61.93	77.01	88.01	85.38	**97.05**	30.39	41.87
MMA w/o DAF	54.65	65.50	88.99	94.05	80.96	93.73	36.07	40.70
MMA w/o CL&DAF	42.39	59.01	73.89	84.47	81.57	96.99	27.92	39.67

Regarding the submodules, MMA achieves optimal performance when all submodules are incorporated. Removing any individual submodule results in a decline in model performance, with the most severe decrease observed when both submodules are removed. The removal of the contrastive learning module results in an average performance loss of 9.56% in VUS-PR and 6.45% in F1-(PA%K). The contrastive learning module generally increases anomaly scores in anomalous regions, making it easier to distinguish between normal and abnormal time points. Additionally, it helps detect subtle anomalies that are difficult to identify solely based on residual errors. Removing the dynamic anomaly filtering module leads to an average performance drop of 3.18% in VUS-PR and 5.17% in F1-(PA%K), indicating that mitigating the influence of anomalous regions on the reconstruction of surrounding normal parts can help reduce false alarms.

4.7 RQ5. Visualization

In Figure 9, we showcase the capability of MMA in detecting various types of anomalies. Some of these anomalies are difficult to detect by other models, such as the Noise-like anomaly in KDD21 003 and the duration change in KDD21 048. Specifically, the Noise-like anomaly, which has a similar amplitude to normal values, can be easily overlooked by methods based solely on residual errors. However, it can still be detected by our contrastive learning module. Some anomalies are challenging even for human experts to identify, such as the rhythmic changes in KDD21 209, where a short beat should follow a long beat. Our model consistently assigns high anomaly scores to these anomalous regions, demonstrating the effectiveness of our proposed method. In addition, our model can also find anomalies in non-stationary time series, such as KDD21 114 and KDD21 245, which contain dynamic changing patterns.

We further provide visualization results to demonstrate the effectiveness of each submodule. Figure 10(a) shows that contrastive learning can aid in detecting subtle anomalies. As these anomalies exhibit similar amplitudes to the reconstructed normal values, "Anomaly Score w/o CL" fails to detect them relying solely on the residual errors. However, "Anomaly Score with CL" can detect them by differentiating them in the embedding space. Figure 10(b) shows that "MMA w/o DAF" fails to accurately reconstruct the normal parts due to the negative impact of the nearby subsequence anomaly. "MMA with DAF" can substitute the anomalous regions with reconstructed normal patterns, resulting in a more precise reconstruction of normal areas and a reduction in false alarms. Figure 10(c) and (d) illustrate that both *MMA_GRU* and *MMA_Grating*

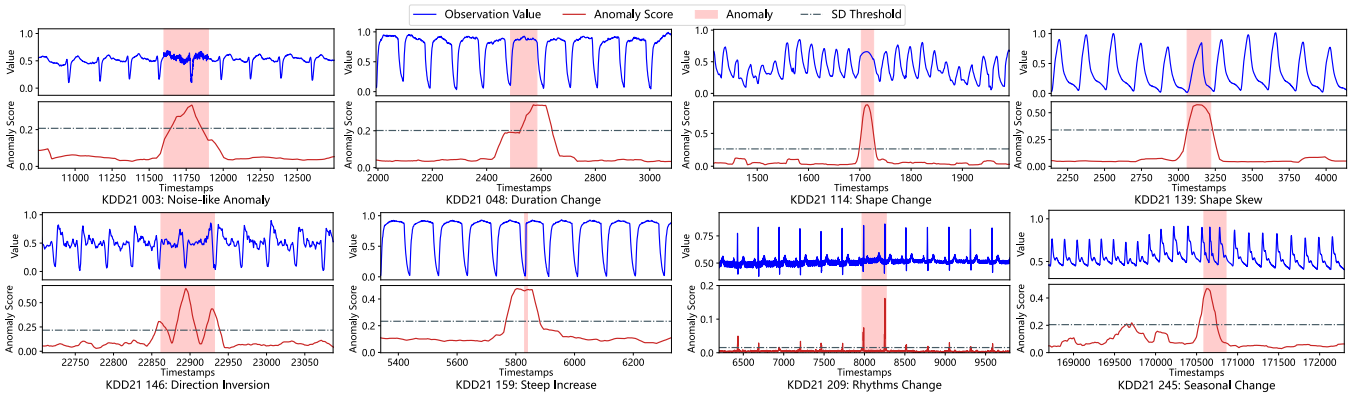


Figure 9: Visualization of the various types of anomalies detected by MMA.

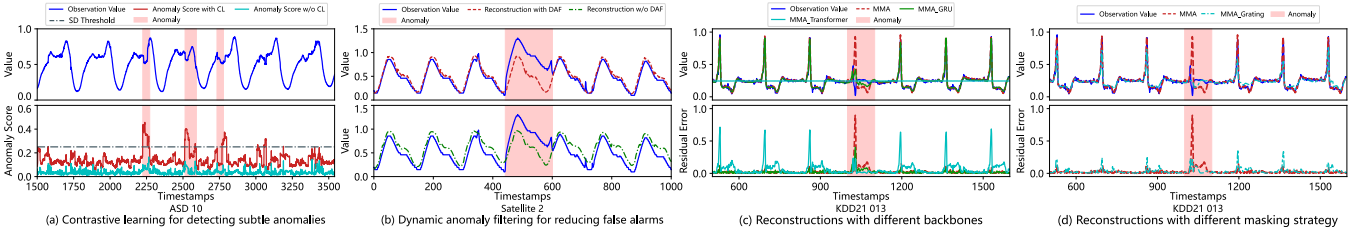


Figure 10: Visualization comparison of the performance of different MMA variants.

tend to overfit anomalies due to their excessive focus on the information surrounding the masked regions. *MMA_Transformer* fails to reconstruct the time series owing to the loss of temporal information. Only MMA is capable of accurately reconstructing normal regions while also restoring normal patterns in anomalous areas.

4.8 RQ6. Insights

4.8.1 Hyperparameters Analysis. The input window size T and the weight λ of the contrastive loss are the most critical hyperparameters in our model. We conduct a sensitive analysis to study the impact of the two parameters. As shown in Figure 11, increasing the input window size significantly enhances MMA's performance. This observation highlights the importance of considering longer contextual information for anomaly detection, particularly for anomalies with long durations. However, when the window size is not large enough, the model will still fail to detect some subsequence anomalies. In such cases, increasing the window size leads to relatively larger residual errors in some normal regions, resulting in false alarms and performance fluctuations, as observed in the ASD and Satellite datasets. Additionally, since the contrastive loss is significantly larger than the reconstruction loss, we set λ to 0.005 to balance the scales of the two losses. As illustrated in Figure 11, both excessively high and low values of λ result in performance losses. When λ is too small, the contrastive learning module is not fully trained, whereas a high λ makes the model more difficult to reconstruct the samples. An λ value ranging from 0.001 to 0.005 yields stable performance across all datasets.

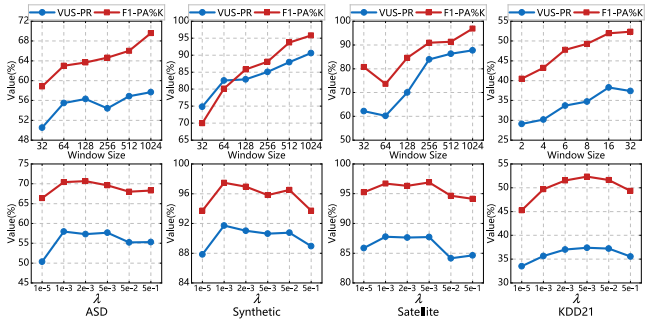


Figure 11: Parameter analysis for the window size and the loss weight λ. The window size of the KDD21 dataset represents multiples of the patch length.

4.8.2 Learned Patch Embeddings. We calculate the pairwise similarity between all patch embeddings within a window and get a 64×64 heat map, where a cell in the i-th column and j-th row represents the cosine similarity between the i-th and j-th patch embeddings. The heat map in Figure 12(a) exhibits a periodicity that aligns with the curves, and the similarities between patch embeddings are highest when the patches are spaced at integer multiples of the period. This proves that our contrastive learning module enables the patch embeddings to capture the shape information of the patches. We also find that in Figure 12(a), the embeddings of patches with similar shapes but different amplitudes also exhibit high cosine similarity, indicating that the patch embeddings are not sensitive to amplitudes. This is an unsolved problem in the field

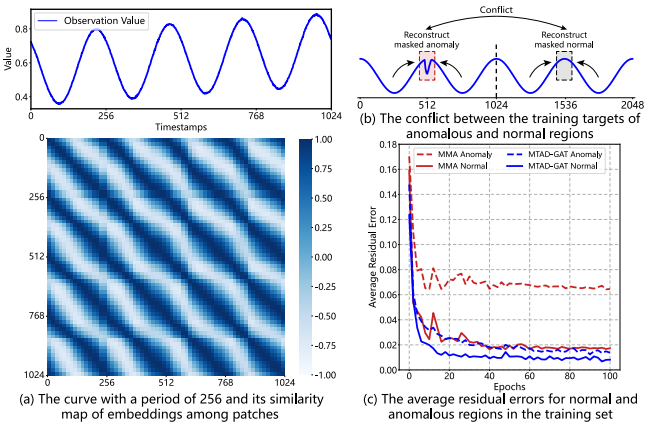


Figure 12: Insights into contrastive learning and robustness.

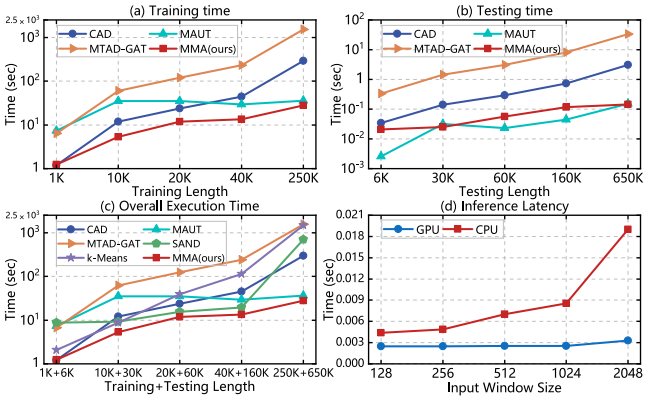


Figure 13: Scalability tests on entities 32, 55, 194, 191, 241 in the KDD21 dataset.

of time series representation learning. Nevertheless, this does not affect our anomaly detection task, as anomalies with large changes in amplitude can still be easily detected through residual errors.

4.8.3 Robustness Analysis. We further analyze why our model demonstrates robustness against anomalies in the training set. As shown in Figure 12(b), when the model is trained to utilize unmasked parts to reconstruct the masked anomalous regions, these regions share similar visible contexts as normal regions but have conflicting reconstruction targets. Since the training set contains significantly more normal samples than anomalies, the model is less likely to be trained to overfit the anomalies. In contrast, commonly used reconstruction-based models map samples into embeddings and then reconstruct them from the embeddings. There is no conflict between the reconstruction of anomalies and normal values, making it easy for the reconstruction-based models to overfit the anomalies. This is further proved in Figure 12(c), where our model and MTAD-GAT (the best performing baseline on the Synthetic dataset) are trained on the Synthetic dataset polluted with real anomalies clipped from the testing set. Our model avoids overfitting the anomalies, consistently maintaining high residual errors in anomalous regions, whereas MTAD-GAT exhibits very small residual errors in anomalous regions after several training epochs.

4.8.4 Scalability Test. We select the top 6 performing models to evaluate their scalability across datasets of varying sizes in KDD21. All the deep learning models are tested on the same NVIDIA RTX 3090 GPU and Non-deep learning models are tested on the same Intel i9-12900K CPU, as they do not support GPU parallel processing. As shown in Figure 13(a) and (b), MMA requires less training and inference time compared to most of the baselines and gracefully scales with the dataset size. This is because MMA allows for larger input windows, enabling it to process more data points in a single forward pass during training or testing. Additionally, the MLP-Mixer backbone used by MMA is proven to be more lightweight than the RNN and Transformer backbones [6, 11]. On the KDD21 241 dataset, MMA requires only a computational load of 37.33M FLOPs and 0.47M Params, while *MMA_GRU* needs 59.38M FLOPs and 0.92M Params, and *MMA_Transformer* demands 69.42M FLOPs and 1.09M Params. Since k-Means and SAND are non-deep learning methods that do not require training, we add the training and testing times of deep learning models as the overall execution time for comparison with them. Figure 13(c) shows that MMA is consistently faster than non-deep learning methods. This is because k-Means and SAND rely on subsequence clustering to detect anomalies, which results in a significant number of pairwise distance computations on large-scale datasets.

In addition, we test MMA's inference latency (response time) to inputs of different window sizes to ensure anomalies are detected promptly in online settings and critical monitoring scenarios. Considering that a powerful GPU is not always available in all deployment environments, we also conduct tests on an Intel x86 2.3GHz CPU. Figure 13(d) shows that MMA requires less than 0.02 seconds to process a large input window of 2048 data points using a CPU. We adopt the commonly used sliding window input method [61] that calculates anomaly scores for a number of new points equal to the sliding stride at each inference step. When MMA takes a sliding stride equal to one period, it can respond to one heartbeat or one engine cycle within 0.02 seconds. This is considered highly efficient in most monitoring scenarios [35]. In addition, reducing the window size or employing a GPU would further decrease the latency substantially.

5 CONCLUSION

This paper presents a novel algorithm, named MMA, which combines the MLP-Mixer backbone with Masked Autoencoders to allow a significantly larger input window for time series anomaly detection. Benefitting from longer contextual information, MMA can detect anomalies with prolonged durations. Additionally, MMA is capable of reconstructing potential normal patterns within anomalous regions, thereby providing high levels of explainability. Furthermore, MMA is robust to anomalies in the training set.

In future research, we will focus on developing a unified model for time series anomaly detection. For example, the KDD21 dataset contains 250 entities, and training a separate model for each entity is time-consuming. We hope to develop a unified model that can handle multiple entities simultaneously.

REFERENCES

[1] Julien Audibert, Pietro Michiardi, Frédéric Guyard, Sébastien Marti, and Maria A. Zuluaga. 2020. USAD:UnSupervised Anomaly Detection on Multivariate Time Series. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, Virtual Event CA USA, 3395–3404. https://doi.org/10.1145/3394486.3403392

[2] Bjorn Barz, Erik Rodner, Yanira Guanche Garcia, and Joachim Denzler. 2019. Detecting Regions of Maximal Divergence for Spatio-Temporal Anomaly Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 5 (May 2019), 1088–1101. https://doi.org/10.1109/TPAMI.2018.2823766

[3] Ane Blázquez-García, Angel Conde, Usue Mori, and Jose A. Lozano. 2022. A Review on Outlier/Anomaly Detection in Time Series Data. *Comput. Surveys* 54, 3 (April 2022), 1–33. https://doi.org/10.1145/3444690

[4] Paul Boniol and Themis Palpanas. 2020. Series2Graph: : Graph-based Subsequence Anomaly Detection for Time Series. *Proceedings of the VLDB Endowment* 13, 12 (Aug. 2020), 1821–1834. https://doi.org/10.14778/3407790.3407792

[5] Paul Boniol, John Paparrizos, Themis Palpanas, and Michael J. Franklin. 2021. SAND: Streaming Subsequence Anomaly Detection. *Proceedings of the VLDB Endowment* 14, 10 (June 2021), 1717–1729. https://doi.org/10.14778/3467861.3467863

[6] Si-An Chen, Chun-Liang Li, Nate Yoder, Sercan O. Arik, and Tomas Pfister. 2023. TSMixer: An All-MLP Architecture for Time Series Forecasting. arXiv:2303.06053 [cs]

[7] Yuhang Chen, Chaoyun Zhang, Minghua Ma, Yudong Liu, Ruomeng Ding, Bowen Li, Shilin He, Saravan Rajmohan, Qingwei Lin, and Dongmei Zhang. 2023. ImDiffusion: Imputed Diffusion Models for Multivariate Time Series Anomaly Detection. *Proceedings of the VLDB Endowment* 17, 3 (Nov. 2023), 359–372. https://doi.org/10.14778/3632093.3632101

[8] Jesse Davis and Mark Goadrich. 2006. The Relationship between Precision-Recall and ROC Curves. In *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*. ACM Press, Pittsburgh, Pennsylvania, 233–240. https://doi.org/10.1145/1143844.1143874

[9] Ailin Deng and Bryan Hooi. 2021. Graph Neural Network-Based Anomaly Detection in Multivariate Time Series. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 5 (May 2021), 4027–4035. https://doi.org/10.1609/aaai.v35i5.16523

[10] Bowen Du, Xuanxuan Sun, Junchen Ye, Ke Cheng, Jingyuan Wang, and Leilei Sun. 2021. GAN-Based Anomaly Detection for Multivariate Time Series Using Polluted Training Set. *IEEE Transactions on Knowledge and Data Engineering* (2021), 1–1. https://doi.org/10.1109/TKDE.2021.3128667

[11] Vijay Ekambaram, Arindam Jati, Nam Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. TSMixer: Lightweight MLP-Mixer Model for Multivariate Time Series Forecasting. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Long Beach CA USA, 459–469. https://doi.org/10.1145/3580305.3599533

[12] Astha Garg, Wenyu Zhang, Jules Samaran, Ramasamy Savitha, and Chuan-Sheng Foo. 2022. An Evaluation of Anomaly Detection and Diagnosis in Multivariate Time Series. *IEEE Transactions on Neural Networks and Learning Systems* 33, 6 (June 2022), 2508–2517. https://doi.org/10.1109/TNNLS.2021.3105827

[13] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollar, and Ross Girshick. 2022. Masked Autoencoders Are Scalable Vision Learners. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, New Orleans, LA, USA, 15979–15988. https://doi.org/10.1109/CVPR52688.2022.01553

[14] Alexis Huet, Jose Manuel Navarro, and Dario Rossi. 2022. Local Evaluation of Time Series Anomaly Detection Algorithms. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Washington DC USA, 635–645. https://doi.org/10.1145/3534678.3539339

[15] Won-Seok Hwang, Jeong-Han Yun, Jonguk Kim, and Byung Gil Min. 2022. Do You Know Existing Accuracy Metrics Overrate Time-Series Anomaly Detections?. In *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing*. ACM, Virtual Event, 403–412. https://doi.org/10.1145/3477314.3507024

[16] Vincent Jacob, Fei Song, Arnaud Stiegler, Bijan Rad, Yanlei Diao, and Nesime Tatbul. 2021. Exathlon: A Benchmark for Explainable Anomaly Detection over Time Series. *Proceedings of the VLDB Endowment* 14, 11 (July 2021), 2613–2626. https://doi.org/10.14778/3476249.3476307

[17] Sevvandi Kandanaarachchi, Mario A. Muñoz, Rob J. Hyndman, and Kate Smith-Miles. 2020. On Normalization and Algorithm Selection for Unsupervised Outlier Detection. *Data Mining and Knowledge Discovery* 34, 2 (March 2020), 309–354. https://doi.org/10.1007/s10618-019-00661-z

[18] Eamonn Keogh. 2021. Irrational Exuberance: Why we should not believe 95% of papers on Time Series Anomaly Detection. Retrieved May 21, 2024 from https://kdd-milets.github.io/milets2021/slides/Irrational%20Exuberance_Eammon_Keogh.pdf

[19] Tung Kieu, Bin Yang, Chenjuan Guo, Razvan-Gabriel Cirstea, Yan Zhao, Yale Song, and Christian S. Jensen. 2022. Anomaly Detection in Time Series with Robust Variational Quasi-Recurrent Autoencoders. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, Kuala Lumpur, Malaysia, 1342–1354. https://doi.org/10.1109/ICDE53745.2022.00105

[20] Tung Kieu, Bin Yang, Chenjuan Guo, Christian S. Jensen, Yan Zhao, Feiteng Huang, and Kai Zheng. 2022. Robust and Explainable Autoencoders for Unsupervised Time Series Outlier Detection. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*. IEEE, Kuala Lumpur, Malaysia, 3038–3050. https://doi.org/10.1109/ICDE53745.2022.00273

[21] HyunGi Kim, Siwon Kim, Seonwoo Min, and Byunghan Lee. 2023. Contrastive Time-Series Anomaly Detection. *IEEE Transactions on Knowledge and Data Engineering* (2023), 1–14. https://doi.org/10.1109/TKDE.2023.3335317

[22] Minkyung Kim, Jongmin Yu, Junsik Kim, Tae-Hyun Oh, and Jun Kyun Choi. 2023. An Iterative Method for Unsupervised Robust Anomaly Detection Under Data Contamination. *IEEE Transactions on Neural Networks and Learning Systems* (2023), 1–13. https://doi.org/10.1109/TNNLS.2023.3267028

[23] Siwon Kim, Kukjin Choi, Hyun-Soo Choi, Byunghan Lee, and Sungroh Yoon. 2022. Towards a Rigorous Evaluation of Time-Series Anomaly Detection. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 7 (June 2022), 7194–7201. https://doi.org/10.1609/aaai.v36i7.20680

[24] Chih-Yu Lai, Fan-Keng Sun, and Zhengqi Gao. 2023. Nominality Score Conditioned Time Series Anomaly Detection by Point/Sequential Reconstruction. In *37th Conference on Neural Information Processing Systems*.

[25] Kwei-Herng Lai, Daochen Zha, Junjie Xu, Yue Zhao, Guanchu Wang, and Xia Hu. 2021. Revisiting Time Series Outlier Detection: Definitions and Benchmarks. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, J. Vanschoren and S. Yeung (Eds.), Vol. 1.

[26] Dongwen Li, Shenglin Zhang, Yongqian Sun, Yang Guo, Zeyu Che, Shiqi Chen, Zhenyu Zhong, Minghan Liang, Minyi Shao, Mingjie Li, Shuyang Liu, Yuzhi Zhang, and Dan Pei. 2023. An Empirical Analysis of Anomaly Detection Methods for Multivariate Time Series. In *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, Florence, Italy, 57–68. https://doi.org/10.1109/ISSRE59848.2023.00014

[27] Gen Li and Jason J. Jung. 2023. Deep Learning for Anomaly Detection in Multivariate Time Series: Approaches, Applications, and Challenges. *Information Fusion* 91 (March 2023), 93–102. https://doi.org/10.1016/j.inffus.2022.10.008

[28] Longyuan Li, Junchi Yan, Qingsong Wen, Yaohui Jin, and Xiaokang Yang. 2022. Learning Robust Deep State Space for Unsupervised Anomaly Detection in Contaminated Time-Series. *IEEE Transactions on Knowledge and Data Engineering* (2022), 1–1. https://doi.org/10.1109/TKDE.2022.3171562

[29] Muyang Li, Xiangyu Zhao, Chuan Lyu, Minghao Zhao, Runze Wu, and Ruocheng Guo. 2022. MLP4Rec: A Pure MLP Architecture for Sequential Recommendations. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, Vienna, Austria, 2138–2144. https://doi.org/10.24963/ijcai.2022/297

[30] Wenkai Li, Cheng Feng, Ting Chen, and Jun Zhu. 2022. Robust Learning of Deep Time Series Anomaly Detection Models with Contaminated Training Data. arXiv:2208.01841

[31] Zhihan Li, Youjian Zhao, Jiaqi Han, Ya Su, Rui Jiao, Xidao Wen, and Dan Pei. 2021. 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*. ACM, Virtual Event Singapore, 3220–3230. https://doi.org/10.1145/3447548.3467075

[32] Zhong Li, Yuxuan Zhu, and Matthijs Van Leeuwen. 2024. A Survey on Explainable Anomaly Detection. *ACM Transactions on Knowledge Discovery from Data* 18, 1 (Jan. 2024), 1–54. https://doi.org/10.1145/3609333

[33] Haoran Liang, Lei Song, Jianxing Wang, Lili Guo, Xuzhi Li, and Ji Liang. 2021. Robust Unsupervised Anomaly Detection via Multi-Time Scale DCGANs with Forgetting Mechanism for Industrial Multivariate Time Series. *Neurocomputing* 423 (Jan. 2021), 444–462. https://doi.org/10.1016/j.neucom.2020.10.084

[34] Shengsheng Lin, Weiwei Lin, Wentai Wu, Feiyu Zhao, Ruichao Mo, and Haotong Zhang. 2023. SegRNN: Segment Recurrent Neural Network for Long-Term Time Series Forecasting. arXiv:2308.11200

[35] Yue Lu, Renjie Wu, Abdullah Mueen, Maria A. Zuluaga, and Eamonn Keogh. 2022. Matrix Profile XXIV: Scaling Time Series Anomaly Detection to Trillions of Datapoints and Ultra-fast Arriving Data Streams. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Washington DC USA, 1173–1182. https://doi.org/10.1145/3534678.3539271

[36] Takaaki Nakamura, Makoto Imamura, Ryan Mercer, and Eamonn Keogh. 2020. MERLIN: Parameter-Free Discovery of Arbitrary Length Anomalies in Massive Time Series Archives. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, Sorrento, Italy, 1190–1195. https://doi.org/10.1109/ICDM50108.2020.00147

[37] Takaaki Nakamura, Ryan Mercer, Makoto Imamura, and Eamonn Keogh. 2023. MERLIN++: Parameter-Free Discovery of Time Series Anomalies. *Data Mining and Knowledge Discovery* 37, 2 (March 2023), 670–709. https://doi.org/10.1007/s10618-022-00876-7

[38] Yuqi Nie, Nam H Nguyen, Phanwadee Sinthong, and Jayant Kalagnanam. 2023. A TIME SERIES IS WORTH 64 WORDS: LONG-TERM FORECASTING WITH TRANSFORMERS. *ICLR* (2023).

[39] John Paparrizos, Paul Boniol, Themis Palpanas, Ruey S. Tsay, Aaron Elmore, and Michael J. Franklin. 2022. Volume under the Surface: A New Accuracy Evaluation Measure for Time-Series Anomaly Detection. *Proceedings of the VLDB Endowment* 15, 11 (July 2022), 2774–2787. https://doi.org/10.14778/3551793.3551830

[40] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S. Tsay, Themis Palpanas, and Michael J. Franklin. 2022. TSB-UAD: An End-to-End Benchmark Suite for Univariate Time-Series Anomaly DetectionTSB-UAD. *Proceedings of the VLDB Endowment* 15, 8 (April 2022), 1697–1711. https://doi.org/10.14778/3529337.3529354

[41] Shuxin Qin, Yongcan Luo, and Gaofeng Tao. 2023. Memory-Augmented U-Transformer For Multivariate Time Series Anomaly Detection. In *ICASSP 2023 - 2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, Rhodes Island, Greece, 1–5. https://doi.org/10.1109/ICASSP49357.2023.10096179

[42] Ferdinand Rewicki, Joachim Denzler, and Julia Niebling. 2023. Is It Worth It? Comparing Six Deep and Classical Methods for Unsupervised Anomaly Detection in Time Series. *Applied Sciences* 13, 3 (Jan. 2023), 1778. https://doi.org/10.3390/app13031778

[43] M. Saquib Sarfraz, Mei-Yen Chen, Lukas Layer, Kunyu Peng, and Marios Koulakis. 2024. Position: Quo Vadis, Unsupervised Time Series Anomaly Detection? arXiv:2405.02678 [cs]

[44] Sebastian Schmidl, Phillip Wenig, and Thorsten Papenbrock. 2022. Anomaly Detection in Time Series: A Comprehensive Evaluation. *Proceedings of the VLDB Endowment* 15, 9 (May 2022), 1779–1797. https://doi.org/10.14778/3538598.3538602

[45] Mohamed El Amine Sehili and Zonghua Zhang. 2023. Multivariate Time Series Anomaly Detection: Fancy Algorithms and Flawed Evaluation Methodology. arXiv:2308.13068 [cs, stat]

[46] Zuogang Shang, Zhibin Zhao, Ruqiang Yan, and Xuefeng Chen. 2023. Core Loss: Mining Core Samples Efficiently for Robust Machine Anomaly Detection against Data Pollution. *Mechanical Systems and Signal Processing* 189 (April 2023), 110046. https://doi.org/10.1016/j.ymssp.2022.110046

[47] Zezhi Shao, Zhao Zhang, Fei Wang, and Yongjun Xu. 2022. Pre-Training Enhanced Spatial-temporal Graph Neural Network for Multivariate Time Series Forecasting. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 1567–1577. https://doi.org/10.1145/3534678.3539396 arXiv:2206.09113 [cs]

[48] Shashi Shekhar, Vagelis Papalexakis, Jing Gao, Zhe Jiang, and Matteo Riondato. 2024. PUPAE: Intuitive and Actionable Explanations for Time Series Anomalies. In *Proceedings of the 2024 SIAM International Conference on Data Mining (SDM)*. Society for Industrial and Applied Mathematics, Philadelphia, PA. https://doi.org/10.1137/1.9781611978032

[49] Yunfei Shi, Bin Wang, Yanwei Yu, Xianfeng Tang, Chao Huang, and Junyu Dong. 2023. Robust Anomaly Detection for Multivariate Time Series through Temporal GCNs and Attention-Based VAE. *Knowledge-Based Systems* 275 (Sept. 2023), 110725. https://doi.org/10.1016/j.knosys.2023.110725

[50] Haotian Si, Changhua Pei, Zhihan Li, Yadong Zhao, Jingjing Li, Haiming Zhang, Zulong Diao, Jianhui Li, Gaogang Xie, and Dan Pei. 2023. Beyond Sharing: Conflict-Aware Multivariate Time Series Anomaly Detection. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE 2023)*. Association for Computing Machinery, New York, NY, USA, 1635–1645. https://doi.org/10.1145/3611643.3613896

[51] Alban Siffer, Pierre-Alain Fouque, Alexandre Termier, and Christine Largouet. 2017. Anomaly Detection in Streams with Extreme Value Theory. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, Halifax NS Canada, 1067–1075. https://doi.org/10.1145/3097983.3098144

[52] Yuting Sun, Guansong Pang, Guanhua Ye, Tong Chen, Xia Hu, and Hongzhi Yin. 2023. Unraveling the "Anomaly" in Time Series Anomaly Detection: A Self-supervised Tri-domain Solution. arXiv:2311.11235 [cs]

[53] Emmanouil Sylligardos, Paul Boniol, John Paparrizos, Panos Trahanias, and Themis Palpanas. 2023. Choose Wisely: An Extensive Evaluation of Model Selection for Anomaly Detection in Time Series. *Proceedings of the VLDB Endowment* 16, 11 (July 2023), 3418–3432. https://doi.org/10.14778/3611479.3611536

[54] Koichi Hori Takehisa Yairi, Yoshikiyo Kato. 2001. Fault Detection by Mining Association Rules from House-keeping Data. In *Proceedings of the International Symposium on Artificial Intelligence, Robotics and Automation in Space (-SAIRAS)*.

[55] Haicheng Tao, Jiawei Miao, Lin Zhao, Zhenyu Zhang, Shuming Feng, Shu Wang, and Jie Cao. 2023. HAN-CAD: Hierarchical Attention Network for Context Anomaly Detection in Multivariate Time Series. *World Wide Web* (May 2023). https://doi.org/10.1007/s11280-023-01171-1

[56] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings. 2022. TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. *Proceedings of the VLDB Endowment* 15, 6 (Feb. 2022), 1201–1214. https://doi.org/10.14778/3514061.3514067

[57] Dennis Wagner, Tobias Michels, Florian C F Schulz, Arjun Nair, Maja Rudolph, and Marius Kloft. 2023. TimeSeAD: Benchmarking Deep Multivariate Time-Series Anomaly Detection. (2023).

[58] Renjie Wu and Eamonn Keogh. 2021. Current Time Series Anomaly Detection Benchmarks Are Flawed and Are Creating the Illusion of Progress. *IEEE Transactions on Knowledge and Data Engineering* (2021), 1–1. https://doi.org/10.1109/TKDE.2021.3112126

[59] Chunjing Xiao, Zehua Gou, Wenxin Tai, Kunpeng Zhang, and Fan Zhou. 2023. Imputation-Based Time-Series Anomaly Detection with Conditional Weight-Incremental Diffusion Models. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ACM, Long Beach CA USA, 2742–2751. https://doi.org/10.1145/3580305.3599391

[60] Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Furtado Silva, Abdullah Mueen, and Eamonn Keogh. 2016. Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, Barcelona, Spain, 1317–1322. https://doi.org/10.1109/ICDM.2016.0179

[61] Jiahao Yu, Xin Gao, Baofeng Li, Feng Zhai, Jiansheng Lu, Bing Xue, Shiyuan Fu, and Chun Xiao. 2024. A Filter-Augmented Auto-Encoder with Learnable Normalization for Robust Multivariate Time Series Anomaly Detection. *Neural Networks* 170 (Feb. 2024), 478–493. https://doi.org/10.1016/j.neunet.2023.11.047

[62] Zhihan Yue, Yujing Wang, Juanyong Duan, Tianmeng Yang, Congrui Huang, Yunhai Tong, and Bixiong Xu. 2022. TS2Vec: Towards Universal Representation of Time Series. *Proceedings of the AAAI Conference on Artificial Intelligence* 36, 8 (June 2022), 8980–8987. https://doi.org/10.1609/aaai.v36i8.20881

[63] Ailing Zeng, Muxi Chen, Lei Zhang, and Qiang Xu. 2023. Are Transformers Effective for Time Series Forecasting? *Proceedings of the AAAI Conference on Artificial Intelligence* 37, 9 (June 2023), 11121–11128. https://doi.org/10.1609/aaai.v37i9.26317

[64] Yuxin Zhang and Yiqiang Chen. 2023. Unsupervised Deep Anomaly Detection for Multi-Sensor Time-Series Signals. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING* 35, 2 (2023).

[65] Zhijie Zhang, Wenzhong Li, Wangxiang Ding, Linming Zhang, Qingning Lu, Peng Hu, Tong Gui, and Sanglu Lu. 2023. STAD-GAN: Unsupervised Anomaly Detection on Multivariate Time Series with Self-training Generative Adversarial Networks. *ACM Transactions on Knowledge Discovery from Data* 17, 5 (Oct. 2023), 1–18. https://doi.org/10.1145/3572780

[66] Hang Zhao, Yujing Wang, Juanyong Duan, Congrui Huang, Defu Cao, Yunhai Tong, Bixiong Xu, Jing Bai, Jie Tong, and Qi Zhang. 2020. Multivariate Time-Series Anomaly Detection via Graph Attention Network. In *2020 IEEE International Conference on Data Mining (ICDM)*. IEEE, Sorrento, Italy, 841–850. https://doi.org/10.1109/ICDM50108.2020.00093

[67] Guoxiang Zhong, Fagui Liu, Jun Jiang, Bin Wang, and C.L. Philip Chen. 2024. Refining One-Class Representation: A Unified Transformer for Unsupervised Time-Series Anomaly Detection. *Information Sciences* 656 (Jan. 2024), 119914. https://doi.org/10.1016/j.ins.2023.119914

[68] Zhijie Zhong, Zhiwen Yu, Yiyuan Yang, Weizheng Wang, and Kaixiang Yang. 2024. PatchAD: A Lightweight Patch-based MLP-Mixer for Time Series Anomaly Detection. arXiv:2401.09793 [cs.LG] https://arxiv.org/abs/2401.09793

[69] Qihang Zhou, Jiming Chen, Haoyu Liu, Shibo He, and Wenchao Meng. 2023. Detecting Multivariate Time Series Anomalies with Zero Known Label. *Proceedings of the AAAI Conference on Artificial Intelligence* 37, 4 (June 2023), 4963–4971. https://doi.org/10.1609/aaai.v37i4.25623