KBJNet: Kinematic Bi-Joint Temporal Convolutional Network Attention for Anomaly Detection in Multivariate Time Series Data

RESEARCH PAPER

MUHAMMAD ABDAN MULIA
MUHAMMAD BINTANG BAHY
MUHAMMAD ZAIN FAWWAZ NURUDDIN SISWANTORO
NUR RAHMAT DWI RIYANTO
NELLA ROSA SUDIANJAYA
ARY MAZHARUDDIN SHIDDIQI

*Author affiliations can be found in the back matter of this article

ABSTRACT

Detecting anomalies in multivariate time series data is crucial to ensure the security and stability of industrial processes. Yet, it remains challenging due to the absence of labeled anomaly data, the complexity of time series data, and the large dataset size. We propose KBJNet, an innovative model incorporating Transformer and Dilated Temporal Convolutional Network (TCN) techniques to address these obstacles. Our model employs a Single TCN-Attention Network, utilizing a single layer of Transformer encoder, making it highly efficient for inference. To further enhance its robustness, we introduce a novel adaptive attention mechanism that dynamically weights temporal context, enabling KBJNet to capture long-range dependencies in time series data effectively. The evaluation of KBJNet on eight publicly available datasets revealed that KBJNet considerably outperforms the most recent methods, enhancing F1 scores by as much as 6%. This result represents a significant contribution to anomaly detection, and we anticipate that our approach will have practical implications for developing next-generation anomaly detection systems in various industrial applications.

CORRESPONDING AUTHOR:
Ary Mazharuddin Shiddiqi
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia
ary.shiddiqi@its.ac.id

KEYWORDS:
anomaly detection; multivariate time series data; Transformer; dilated convolution

TO CITE THIS ARTICLE:
Abdan Mulia, M, Bahy, MB, Siswantoro, MZFN, Riyanto, NRD, Sudianjaya, NR and Shiddiqi, AM. 2024. KBJNet: Kinematic Bi-Joint Temporal Convolutional Network Attention for Anomaly Detection in Multivariate Time Series Data. *Data Science Journal*, 23: 10, pp. 1–22. DOI: https://doi.org/10.5334/dsj-2024-010

I. INTRODUCTION

Nowadays, IT activities generate a significant amount of high-dimensional sensor data. Although big data analytics and deep learning have made handling massive amounts of data possible, identifying irregularities in such data remains challenging due to the vast volume, noise, and uneven data distribution that make it difficult to detect anomalies. This phenomenon is called the 'dimensionality curse' (Thudumu et al. 2020). Moreover, anomalies can arise from interactions between multiple causes, which further complicates the detection process. This problem domain is particularly crucial in data-driven industries that generate many unstable, dispersed, and multimodal time series datasets, such as source management, autonomous driving, and the Internet of Things (IoT).

Anomalies reveal unusual characteristics within the systems and entities responsible for supplying data. These atypical traits offer valuable insights for real-world applications. Detecting data anomalies can uncover outliers, identify environmental conditions requiring human attention, or optimize computing resources by preemptively filtering undesired data segments. For cloud systems, promptly identifying anomalies following an incident is crucial in preventing more significant failures that may impact customers (Darban et al. 2022). The research also explained that intrusion detection plays a vital role in computer network systems by distinguishing between illegal and malicious behaviors. Another aspect that the research covered was the electrocardiography (ECG) signals for assessing heart conditions in medicine. Typically, medical practitioners manually evaluate the resulting time series signal to detect arrhythmia. Finally, a multivariate industrial time series monitors these processes, incorporating data from sensors and control systems within the gas-oil plant heating loop (GHL). An LSTM-based technique is used to detect defects in this context.

Anomaly detection involves identifying data points, patterns, or traffic that significantly deviate from a system's expected behavior. Outliers that deviate substantially from the rest of the distribution are labeled as anomalies (Bulusu et al. 2020). Anomaly detection is essential for creating trustworthy computer systems (Wang et al. 2022) in commercial, industrial, healthcare, and military applications to ensure crucial processes or decisions are safe (Sarker 2021). Anomaly detection based on statistical, rule-based, machine learning, and neural networks with unsupervised methods is becoming increasingly important. These methods provide fast inference speed, improve quality of service, and efficiently manage high-dimensional time series data (Chatterjee & Ahmed 2022).

Various statistical, rule-based, and machine-learning methods have previously been developed to find abnormalities in time series data. Rule-based methods compare data to an anomaly rule, which can be flawed and require frequent updates, making it time-consuming. Statistical methods estimate parameters based on a particular distribution but may fail to capture underlying nonlinearities and dynamical linkages. Machine learning approaches come in three types: supervised, unsupervised, and weakly supervised learning. Unsupervised techniques such as One-Class Support Vector Machine (OC-SVM) (Schölkopf et al. 2001), k-Nearest Neighbor (KNN) (Ramaswamy et al. 2000), Support Vector Data Description (SVDD) (Tax & Duin 2004), Expectation Maximization (EM) (Pan et al. 2010), Histogram-Based Outlier Score (HBOS) (Goldstein & Dengel 2012), Local Outlier Factor (LOF) (Breunig et al. 2000), and Local Density Cluster-based Outlier Factor (LDCOF) (Amer & Goldstein 2012) have already been employed for identifying anomalies in time series data. However, they may have issues in capturing temporal correlation and performance. Statistical methods such as wavelet theory, Hilbert transform (Chowdhury et al. 2017), principal component analysis (PCA) (Jin et al. 2017), and Markov chain models (Zang et al. 2018) has also been used for time series data analysis. Recently, machine learning methods such as SVM (Budiarto et al. 2019), Regression models (Hu et al. 2020), and clustering (Budiarto et al. 2019) have been created to forecast the distribution of time series data. However, memory constraints can limit their ability to detect temporal patterns.

Anomaly detection methods using deep learning have attracted interest and become popular due to their ability to handle challenging detection problems in various real-world applications. Recurrent neural networks (RNNs) can be a good option to solve sequence modeling problems. However, traditional RNNs struggle to capture remote relationships

due to gradient disappearance in long-sequence modeling problems. Popular RNN (Shih et al. 2019) variations, including gated recurrent unit (GRU) (Qu et al. 2018) and long short-term memory (LSTM) (Provotar et al. 2019) have already been created to get around this restriction. In modeling temporal patterns, RNNs can benefit from the attention mechanism. However, the computational intensity and slow speed of recursive models such as LSTM hinder their ability to replicate long-term trends accurately. In contrast, some time-series anomaly detection tasks, such as detecting anomalies in sensor data or financial transactions, may require detecting subtle deviations from normal behavior over long periods. The dual-path network has been proposed as an effective method to solve this problem (Luo et al. 2020).

Recently, the Transformer model's encoding of large sequences allows for almost independent accuracy and inference time, making it an excellent choice for anomaly detection models that mine long-term dependencies and deal with nonlinear dynamics. Nonetheless, the Transformer model can only handle sequences with a length of a few hundred (Chen et al. 2020). The Transformer model has a significant computational complexity for extended sequences, and the training is slow. To address these issues, recent research has proposed combining temporal convolution networks (TCN) with transformers to capture temporal dependencies while avoiding the pitfalls of recursive models (Yin et al. 2022).

While there have been notable improvements in anomaly detection for time series data, conventional statistical approaches and machine learning algorithms have limitations in effectively handling nonlinear, high-dimensional, and noisy data. Although LSTM and GRU neural networks can capture contextual information, they face challenges due to their slow inference speed and inefficiency. On the other hand, transformers demonstrate strengths in parallelization and capturing long-range dependencies in input sequences. However, slow training and high computational complexity hinder their performance on longer sequences.

Based on the aforementioned considerations, we introduce a novel model called KBJNet, which integrates the TCN and transformers architectures using a dual-path network for detecting abnormalities in multivariate time series data. The KBJNet model incorporates an adaptable multi-head mechanism for attention that comprehensively captures the characteristics of each dimension in the data, enabling effective anomaly detection. Our key contributions include:

- Our study proposes a new model architecture for capturing anomalies involving a combination of dilated TCN and transformers. The TCN utilizes dilation convolution to establish a perceptual field. To ensure a global perceptual field that covers the whole input sequence, the minimum number of convolutional layers is determined based on factors such as the input sequence length, convolution kernel size, and dilation coefficient. In other words, the range of the dilation convolution is adjusted to encompass the entire input sequence.
- We embed this combined TCN and transformers into a dual-path network, which enhances its efficiency and effectiveness for modeling extremely long sequences and high dimensions.
- We introduce a dual path network that utilizes a shared TCN Attention mechanism for assigning weights to time steps. This approach facilitates recognizing and prioritizing crucial information within a multivariate time series.
- Our method has undergone comprehensive testing on standard datasets and has demonstrated superior performance compared to the current leading techniques in benchmark tests.

II. LITERATURE REVIEW

This section presents a comprehensive literature review on anomaly detection, emphasizing three crucial areas: statistical and machine learning approaches, neural network and deep learning techniques, and the current state-of-the-art. Table I summarizes terminologies used in this study.

DOI: 10.5334/dsj-2024-010

TERMINOLOGY	DEFINITION
ARIMA	Autoregressive Integrated Moving Average
AUC	Area under the ROC Curve.
CAV	Connected and Autonomous Vehicle
COPOD	Copula-Based Outlier Detection
CPOD	Core Point-based Outlier Detection
DAGMM	Deep Autoencoding Gaussian Mixture Model
DTAAD	Dual Tcn-Attention Networks for Anomaly Detection in Multivariate Time Series Data
ECG	Electrocardiography
EVT	Extreme Value Theory
FFN	Feedforward Neural Network
GAN	Generative Adversarial Network
GDN	Graph Deviation Networks
GHL	Gas-oil Plant Heating Loop
GPD	Generalized Pareto Distribution
GRU	Gated Recurrent Unit
GTA	Graph Learning with Transformer for Anomaly Detection
HBOS	Histogram-Based Outlier Score
IoT	Internet of Things
KBJNet	Kinematic Bi-Joint Temporal Convolutional Network Attention for Anomaly Detection
KDD	Knowledge Discovery and Data Mining
KNN	k-Nearest Neighbor
LDCOF	Local Density Cluster-based Outlier Factor
LOF	Local Outlier Factor
LSTM	Long Short-Term Memory Networks
LSTM-VAE	Long Short-Term Memory Networks and Variational Autoencoder
MAD-GAN	Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks
MAML	Model-Agnostic and Meta-Learning
MBA	MIT-BIH Supraventricular Arrhythmia Database
MSCRED	Multi-Scale Convolutional Recurrent Encoder-Decoder
MSDS	Material Safety Data Sheet
MSE	Mean Squared Error
MSL	Mars Science Laboratory
MTAD-GAT	Multivariate Time-Series Anomaly Detection via Graph Attention Networks
MTS	Multivariate Time Series
NAB	Numenta Anomaly Benchmark
NSIBF	Neural System Identification and Bayesian Filtering
PCA	Principal Component Analysis
POT	Peaks Over Threshold
ReLU	Rectified Linear Unit
RNN	Recurrent Neural Network
SMAP	Soil Moisture Active Passive
SMD	Server Machine Dataset
SoTa	State of the Art
SVD	Support Vector Data
SVDD	Support Vector Data Description
SVM	Support Vector Machine
SWaT	Secure Water Treatment
TCN	Temporal Convolutional Network
TranAD	Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data
TWSVM	Twin Support Vector Machine
USAD	Unsupervised Anomaly Detection
UTRAD	Anomaly Detection and Localization with U-Transformer
WADI	Water Distribution

Table I Summary of terminology used.

A. STATISTICAL AND MACHINE LEARNING

Several commonly used time series anomaly detection techniques include 3sigma, PCA, KNN, copula-based outlier detection (COPOD), LOF, and OC-SVM. The 3sigma method measures deviations from historical averages, while PCA calculates eigenvector distance differences according to Shyu et al. (2003). KNN determines anomalies based on the mean distance of nearest neighbors, as discussed in Kiss et al. (2014). COPOD utilizes statistical probability functions, OC-SVM seeks to learn decision boundaries for typical observations, and LOF is an unsupervised method based on density, as described by Li et al. (2020).

Patcha & Park (2007) introduced an outline of several methods for anomaly detection, including hidden Markov chains, PCA, process regression, and isolation forest, while also highlighting their limitations. Yaacob et al. (2010) introduced an auto-regressive integrated moving average (ARIMA) method, as a representative statistical approach for modeling and detecting anomalous behaviors. Bandaragoda et al. (2014) widely used isolation forest, which recursively divides the feature space using multiple isolation trees for anomaly detection.

In the healthcare sector, Salem et al. (2014) utilized linear regression combined with SVM to capture anomaly detection in wireless sensor networks. Shang et al. (2018) introduced SVM combined with mean clustering to increase the effectiveness of model training and enhance anomaly detection precision. Boniol et al. (2020) presented GraphAn, a graph-based approach that converts time series data using interval graph distance. Tran et al. (2020) employed clustering and database manipulation history in their outlier detection method, called CPOD. Kingsbury & Alvaro (2020) proposed Elle, another outlier detection method that leverages clustering and database manipulation history.

In their study, Dhiman et al. (2021) employed adaptive threshold techniques and Twin Support Vector Machines (TWSVM) to detect anomalies within two univariate time series data. They proposed these methods as effective approaches in their study. On the other hand, Wang et al. (2021) focused on enhancing the security of CAV (connected and autonomous vehicle) systems. They used an adaptive extended Kalman filter with a pre-trained single-class SVM. Their strategy attempted to increase the CAV systems' overall security.

B. NEURAL NETWORK AND DEEP LEARNING

Several deep learning-based methods have already been proposed to resolve it. For robust anomaly detection, LSTM-based neural network architecture is used by neural system identification and Bayesian filtering (NSIBF) for Bayesian filtering and system identification. EncDec-AD (Malhotra et al. 2016) used LSTM as the base cell for both the encoder and decoder. To recreate the error for each input data and produce a representation with low-dimension, deep autoencoding Gaussian mixture model (DAGMM) (Zong et al. 2018) uses a deep autoencoder. The advantage of this method is it will not exploit temporal information. Meanwhile, MSCRED (Zhang et al. 2019) uses a convolutional encoder-decoder and an attention-based Conv-LSTM to recreate a multi-scale signature matrix. This will use residual signature matrices to detect anomalies, but it may take longer training time and limited performance with insufficient data.

Ergen and Kozat (2020) introduced an algorithm that uses LSTM to transform dynamic data length sequences into sequences with static length, then a single-class support vector machine-based anomaly detector decision function or a support vector data description technique comes next. OmniAnomaly (Su et al. 2019) proposed a recurrent neural network incorporating stochasticity to identify irregularities in multivariate time series data. LSTM-VAE (Park et al. 2018) combined LSTM and variational autoencoder but overlooked the interconnection between stochastic variables. Multivariate anomaly detection for time series data with generative adversarial networks (MAD-GAN) (Li et al. 2019) adopts generator and discriminator base models in the GAN framework that utilizes LSTM-RNN to visualize time series distributions' temporal relations. TCN-AE (Thill et al. 2021) ignores the correlation between time series and combines TCN and AE. Multivariate time-series anomaly detection via GAN (MTAD-GAT) (Zhao et al. 2020) employs GAT (GATs) (Veličković et al. 2018) in both the feature and time dimensions to capture temporal and feature correlations. Anomaly Transformer (Xu et al. 2022)

proposed a minimax training strategy and used self-attention weights to identify anomalies. Graph learning with transformer for anomaly detection (GTA) (Chen et al. 2022) employed an architecture based on transformers to learn and capture temporal dependencies. They utilized this approach to acquire a graph structure that accurately represents the relationships between different elements within the data. Deep transformer networks for anomaly detection (TranAD) (Tuli et al. 2022) incorporated adversarial training and self-conditioning techniques in a transformer-based model to improve performance.

Huang et al. introduced HitAnomaly, an anomaly detection model based on log analysis. HitAnomaly utilizes a hierarchical transformer structure to capture and represent both the sequences of log templates and their corresponding parameter values. The classification model developed by the researchers was constructed by incorporating an attention mechanism. Additionally, they devised separate log sequences and parameter value encoders to obtain their respective representations. The study provides evidence that the transformer model outperforms LSTM and illustrates the successful modeling of log sequences using a hierarchical framework. Using three log datasets, the results demonstrated that Other currently used log-based anomaly detection methods have not performed as well as HitAnomaly (Huang et al. 2020).

Yu et al. (2023) combines autoregressive (AR) and adaptive ensemble (AE) with the addition of the transformer to capture the information of long sequences. Design convolution and dilated convolution as local TCN, introduce feedback mechanism, and loss ratio to improve detection accuracy and expand association differences.

C. STATE OF THE ART

The deep learning methodology has promising performance in multivariate time series (MTS) anomaly detection. Various approaches, including transformer-based models, autoencoder-based models, and others, have been proposed, each with unique architectures and techniques. These models represent substantial progress in MTS anomaly detection and offer enticing possibilities for future research endeavors. However, a notable challenge in deep learning methodologies is the slow training process and the considerable computational complexity, potentially hindering their efficacy, particularly when dealing with longer sequences. We summarize the features of the state-of-the-art methods in Table II, highlighting the capabilities of our proposed method.

Table II Summary of literature review multivariate time series.

METHOD	APPROACH	MAIN ARCHITECTURE	SUPERVISED/ UNSUPERVISED	ABLE TO HANDLE LIMITED DATA	INTERPRETABILITY
DAGMM (Zong et al. 2018)	Forecasting	AE	Unsupervised	×	×
HitAnomaly (Huang et al. 2020)	Forecasting	Transformer	Supervised	×	×
TCN-AE (Thill et al. 2021)	Reconstruction	AE	Unsupervised	×	×
OmniAnomaly (Su et al. 2019)	Reconstruction	VAE	Unsupervised	×	×
LSTM-VAE (Park et al. 2018)	Reconstruction	VAE	Semi	×	×
GTA (Chen et al. 2022)	Reconstruction	GNN	Semi	×	×
MSCRED (Zhang et al. 2019)	Reconstruction	AE	Unsupervised	×	✓
MAD-GAN (Li et al. 2019)	Reconstruction	GAN	Unsupervised	×	×
USAD (Li et al. 2019)	Reconstruction	AE	Unsupervised	×	×
MTAD-GAT (Zhao et al. 2020)	Hybrid	GNN	Supervised	×	✓
CAE-M (Zhang et al. 2021)	Hybrid	AE	Unsupervised	×	×
GDN (Deng & Hooi 2021)	Forecasting	GNN	Unsupervised	×	✓
TranAD (Tuli et al. 2022)	Reconstruction	Transformer	Unsupervised	✓	✓
DTAAD (Yu et al. 2023)	Reconstruction	Transformer	Unsupervised	✓	✓
KBJNet	Reconstruction	Transformer	Unsupervised	✓	✓

III. METHODOLOGY

In this section, we present a comprehensive methodology for addressing the problem formulation of anomaly detection using a combination of advanced machine learning techniques. Our methodology encompasses various stages, including data preprocessing, the implementation of dilated temporal convolutional networks (TCN), transformers, and a novel kinematic bi-joint TCN and transformer model. We also describe the training, meta-learning techniques, and inference procedures for efficient anomaly detection and diagnosis. Furthermore, we provide a summary of the performance measures employed to assess the efficiency of our approach in detecting anomalies. By integrating these components, our methodology offers a resilient and precise solution for identifying anomalies in real-world applications.

A. PREPROCESS

We examine a set of data points or observations organized in a time-stamped sequence and numerous variables. Each datapoint in the set T is gathered at a unique timestamp t, forming the datapoints x_t of the set T. Each x_t belongs to the vector space of real numbers with dimension m, for all values of t. In the univariate setting, $m = 1$. We assume that the joint probability of the entire time series $\mathbf{x}$ can be factorized into a product of conditional probabilities, where each observation at time t is conditionally dependent on the past observations $x_1^{(i)}, x_2^{(i)}, \ldots, x_{t-1}^{(i)}$ in the same time series component i.

Given a multivariate time series input as the sum of values $z_{i,1:t_0}^l$ for each time series i and dimension l. Each $z_{i,1:t_0}^l$ represents a sequence of values $z_{i,1}^l, z_{i,2}^l, \ldots, z_{i,1:t_0}^l$ in the l-th dimension of the time series data, where $z_{i,1:t_0}^l$ is a vector in $\mathbb{R}^m$. Each data point $x_t^{(i)}$ is a vector in $\mathbb{R}^m$. To increase training stability and strengthen the resilience of KBJNet, we take steps to standardize datasets obtained from different sources.

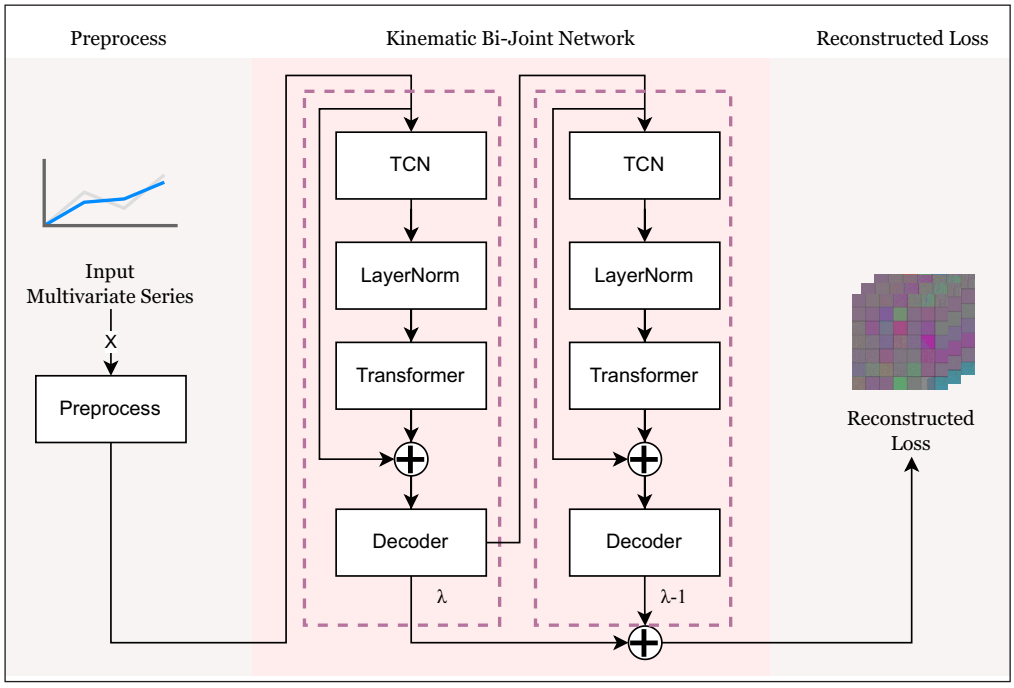

Figure 1 Kinematic bi-joint network architecture for anomaly detection.

In the data preprocessing stage, we filter out nonessential information from the datasets to concentrate only on the crucial data for anomaly detection. We exclude irrelevant details such as the source and description of the dataset and other unnecessary information. Instead, we emphasize essential elements like the dataset size, anomaly labels, and the time steps. Additionally, we standardize the data formats and specifications to ensure consistency throughout the dataset.

The data is normalized and transformed into time-series windows for training and testing. The normalization of the time-series data is conducted by applying the following equation:

$$x_t \leftarrow \frac{x_t - \min(\mathcal{T})}{\max(\mathcal{T}) - \min(\mathcal{T}) + \epsilon'}, \tag{1}$$

B. SLIDING WINDOW

To represent the relationship of a value x_t in a specific timestamp t, we investigate a relevant window of a certain length K as

$$W_t = \{x_{t-K+1}, \ldots, x_t\} \tag{2}$$

For timestamps less than K, to incorporate replication padding, we extend the window W_t by adding a constant vector of length K-t. The input time series $\mathcal{T}$ is then converted into a sequence of sliding windows $\mathcal{W} = \{W_1, \ldots, W_T\}$. The use of sliding windows with replication padding helps preserve the data points' local context, as shown in Figure 2.

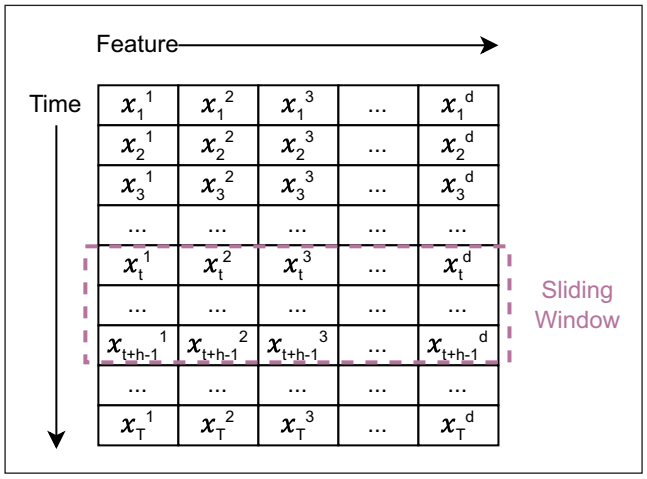

Figure 2 An illustration or depiction of data that involves multiple variables and occurs over a period of time.

W_t and O_t, the anomaly score s_t is computed.

The input window is labeled anomalous if its anomaly score is greater than the threshold value, which is calculated using the anomaly scores of the previous input windows.

C. DILATED TCN

We have developed a novel architecture to enhance feature-sharing efficiency while retaining the network's ability to learn new features. Our approach involves implementing a bi-joint TCN design in which all blocks share a common dilated TCN. This approach significantly reduces redundancy in the feature extraction process while enabling the network to learn new features through its densely connected path.

The dilated convolution operation, concluded in Figure 3, is used in convolutional neural networks, known as a jump filter, that expands the receptive field exponentially in each layer. For a 1-D sequence input $x \in \mathbb{R}^n$ and a convolutional filter $f = \{0, \ldots, k-1\} \in \mathbb{R}$, the operation F on an element s of the sequence is defined as

$$\mathcal{F}(s) = (x *_d f)(s) = \sum_{i=0}^{k-1} f(i) \cdot x_{s-d \cdot i} \tag{3}$$

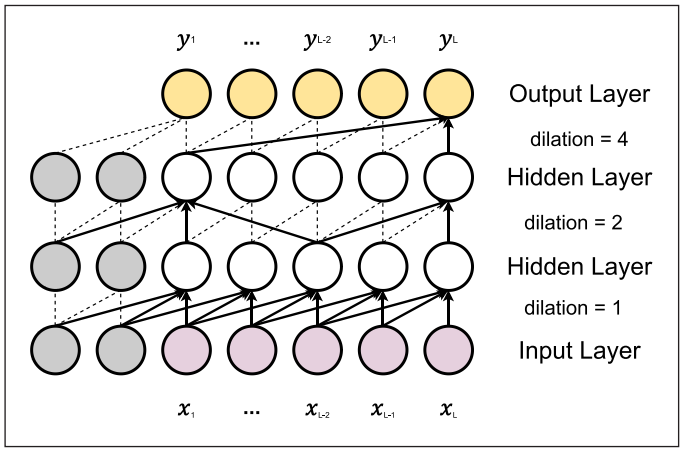

Figure 3 The convolution has specific dilation factors of 1, 2, and 4 and a kernel size of 3. The input is represented as x, and the output is represented as y.

where d denotes the dilation factor, k is the convolutional filter size, and $s - d \cdot i$ indicates the index to the past according to d. In general, the receptive field r of a 1D convolutional network with n layers and a kernel size of k is given by $r = 1 + n \cdot (k - 1)$. To completely cover the input length, we set the number of layers n such that $n = \lceil (l-1)/(k-1) \rceil$, where $\lceil \cdot \rceil$ is rounded up. However, this causes the network to become too deep, resulting in a model with many parameters. We obtain a minimum number of layers required by the global TCN (Yu et al. 2023).

Our proposed approach involves feeding the decoder output back into the same TCN for additional processing, which helps the model improve the input data representation over time. This process potentially captures more complex patterns. The feedback loop between the decoder facilitates the model's learning and adjustment to the input data.

D. TRANSFORMER

The Transformer model, widely used in natural language processing and machine vision, is based on attention. Attention scoring computes the dot product of d_k-dimensional queries and keys and the d_v-dimensional value, then applies a softmax activation function to the result to obtain weights multiplied by the value. This scoring function is efficient and compact. In the transformer, inputs undergo a transformation process, creating query, key, and value matrices Q, K, and V. To simplify the subsequent neural network model inference operations, the matrix $\mathbf{V}$ is compressed into a smaller representative embedding space using the softmax distribution to generate convex combination weights. The square root of the $\sqrt{d_k}$ is used to stabilize the model's gradient, reduce weight fluctuations, and promote more stable training.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{4}$$

where Q, K, and V are matrices in $\mathbb{R}^{n \times d_{\text{model}}}$, and d_{model} is a learned dimension. Multi-headed attention enables the model to focus on diverse information simultaneously, and the result is concatenated and transformed using a linear projection to obtain d_{model}-dimensional features. The model consists of two encoders and one decoder, with position encoding added to the output of the model's first half to obtain the encoders' input.

Position encoding is performed using sine and cosine functions where *pos* is the token's position in the sequence, i is the index of the dimension in the encoding, and d_model is the dimension of the model. The FFN layers apply two linear layers with leaky ReLU activation functions to the input data. The first FFN's output was then routed through the second linear layer to generate the FFN's final output. In the decoder, the last FFN is then passed through by a sigmoid activation function.

E. KINEMATIC BI-JOINT TCN AND TRANSFORMER

The kinematic bi-joint TCN and transformer, as concluded in Figure 1 model processes input from a dilated TCN with dimensions (B, L, C), where B is the batch size, and L is the sequence length, and C is the number of features. The input is normalized using LayerNorm, which calculates the mean (μ) and variance (σ^2) along the feature dimension as follows:

$$\mu = \frac{1}{L}\sum_{l=1}^{L} X_{blc} \tag{5}$$

$$\sigma^2 = \frac{1}{L}\sum_{l=1}^{L} (X_{blc} - \mu)^2 \tag{6}$$

The normalized input $\hat{X}_{blc}$ at position (b, l, c) is obtained by subtracting μ from X_{blc} and dividing by the square root of $\sigma^2 + \epsilon$, where ϵ is a small constant added for increasing numerical stability:

$$\hat{X}_{blc} = \frac{X_{blc} - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{7}$$

The normalized tensor is then adjusted by scaling and shifting using γ_c and β_c learnable parameters to get the output Y_{blc} of the LayerNorm operation at position (b, l, c):

$$Y_{blc} = \gamma_c \hat{X}_{blc} + \beta_c \tag{8}$$

Both γ_c and β_c are learnable parameters updated during training. The sliding window output $\vec{T}$ is then transferred to a stack of *B* bi-joint TCN transformer blocks.

Each bi-joint block part of our model comprises one transformer encoder and one decoder. We then combine the output of the first part of the model with position encoding to obtain the input I_i, which is then passed through two separate encoders:

$$I_i^1 = \textit{Layer-Norm}(I_i + \textit{MultiHead}(I_i, I_i, I_i)) \tag{9}$$

$$I_i^1 = \textit{Layer-Norm}(I_i + \textit{MultiHead}(\vec{T}_b, \vec{T}_b, \vec{T}_b)) \tag{10}$$

where $i \in \{1, 2\}$ for the first and second encoder. The encoder's output is then connected to the feedforward layer using residual connections and sent separately to the two decoders to obtain the final predicted outputs:

$$I_i^3 = I_i^2 + \textit{FFN}_1(\text{LeakyReLU}(\textit{FFN}_2(I_i^2))) \tag{11}$$

$$\mathcal{O}_i = \textit{Sigmoid}(\textit{FFN}(I_i^3)) \tag{12}$$

The sigmoid activation function is used to constrain the output range of $\mathcal{O}_i$ to be between 0 and 1, which is suitable for the later error reconstruction with the normalized sliding window input.

F. PROCEDURE FOR TRAINING

We use mean squared error (MSE) as the loss criterion to measure the error between the output prediction of each decoder and the original input window x_t. We calculate the losses of the two decoders as L_1 and L_2, respectively, using the following equations:

$$L_1 = \frac{1}{n}\sum_{i=1}^{n}(O_1 - x_i)^2, \quad L_2 = \frac{1}{n}\sum_{i=1}^{n}(O_2 - x_i)^2 \tag{13}$$

To obtain the total loss $\mathcal{L}$, we combine the losses of the two decoders from the first TCN and the second TCN by taking a weighted sum with a hyperparameter λ. The goal is to minimize the total loss of the hyperparameters W and model parameters Θ:

$$\{\Theta^*, W^*\} = \arg\min_{\Theta, W} \sum_{x \in \mathcal{X}} \mathcal{L}(\psi(\phi(x; \Theta); W)) \tag{14}$$

where ϕ represents the overall network with total model parameters Θ, W denotes the collection of hyperparameters, and ψ represents the overall learning mapping for anomaly detection task.

G. META LEARNING

To improve the training of our KBJNet model with limited data, which exists in Algorithm 1 line 12, In every training epoch, we update the weights of neural networks θ with a gradient descent step using the loss function *L* and the learning rate α.

Encoders E_1 and E_2, Decoders D_1 and D_2
Sliding windows $\mathcal{W}$
Split time series into the dataset $\mathcal{D}$
Hyperparameters W
Iteration N
1: Randomly initialize Θ_e, Θ_d
2: $n \leftarrow 0$
3: **While** $n < N$
4: **for** $t = 1$ to $\mathcal{W}$ **do**
5: Compute $\mathcal{O}_1$
6: Compute $\mathcal{O}_2$, $\mathcal{O}_2 \leftarrow \varphi_\phi^{D_2}\left(\phi\left(\mathcal{X}_{:,t} + \mathcal{O}_1; \Theta_{E_1}\right); \mathrm{W}\right)$
7: Calculate and combine loss two decoder
8: Calculate gradient
9: Update $\{\Theta_e, \Theta_d\}$
10: $n \leftarrow n + 1$
11: Learn meta weights E_1, E_2, D_1, D_2

Algorithm 1 The KBJNet Training Algorithm.

This gives us the updated weights θ'. Model-agnostic and meta-learning (MAML) (Finn et al. 2017) is performed at the end of each epoch using the updated weights to update the model parameters θ with a meta step-size β. As a result, the model can be trained quickly with limited data. The algorithm can be written as:

$$\theta' \leftarrow \theta - \alpha \nabla_\theta L(f(\theta)), \quad \theta \leftarrow \theta - \beta \nabla_\theta L(f(\theta')) \tag{15}$$

H. INFERENCE PROCEDURE, ANOMALY DETECTION, AND DIAGNOSIS

Our approach, as concluded in Algorithm 2, involves performing online inference sequentially on a sliding window of input data, generating anomaly scores for each timestamp in each dimension. The Peak Over Threshold (Siffer et al. 2017) approach is used to dynamically select thresholds for each dimension by applying the Extreme Value Theory (EVT) to the univariate time series of anomaly scores obtained during offline training. Instead of manually setting thresholds and making assumptions about the distribution, we use the Generalized Pareto Distribution (GPD) (Siffer et al. 2017) function following EVT to fit the data and determine the appropriate value-at-risk (label) for dynamically setting the threshold, which is consistent with OmniAnomaly (Su et al. 2019), TranAD (Tran et al. 2020), and DTAAD (Yu et al. 2023) (Figure 4).

Require:
Encoders E_1 and E_2, Decoder D_1
Sliding window size $\hat{W}$
Split time series into dataset $\hat{D}$
Hyperparameters W
1: Trained models $\hat{\Theta}_e$, $\hat{\Theta}_d$
2: Randomly sample one batch from dataset $\hat{D}$
3: **for** $t = 1$ to $\hat{W}$ **do**
4: Compute $\hat{O}_1 \leftarrow \varphi_\phi^{D_1}(\phi(X_{:,t}; \hat{\Theta}_{E_1}); W)$
5: Calculate loss s_i
6: Merge the exceptions D, from y_i $(s_i \geq \text{POT}(s_i))$
7: **return** D;

Algorithm 2 The KBJNet Testing Algorithm.

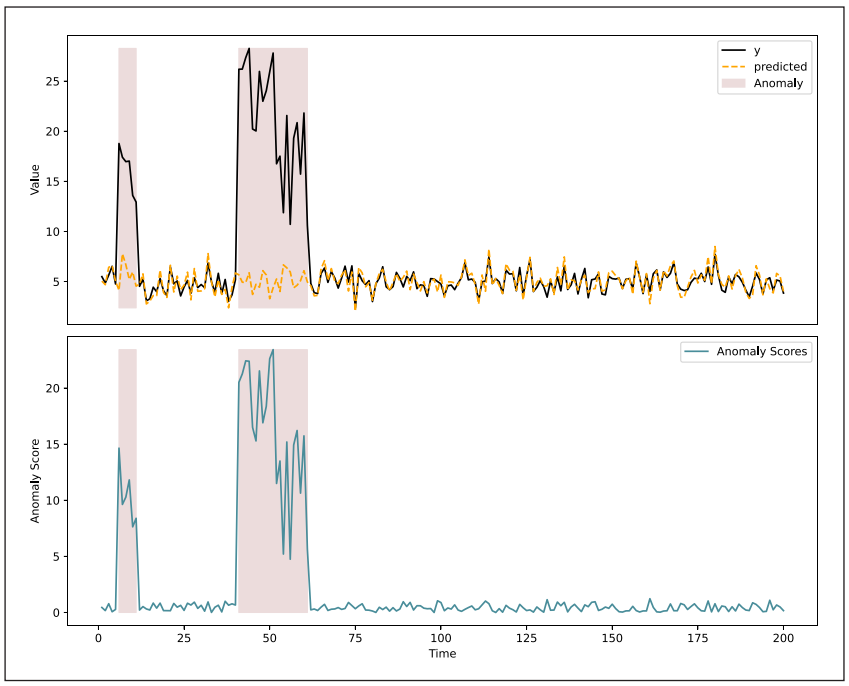

Figure 4 Visualization of anomaly prediction.

IV. EXPERIMENTS

We did tests to assess the effectiveness of our model, KBJNet. The dataset used in our experiments, as well as the performance metrics used, are described. We compared KBJNet with the most widely used models and advanced methods currently available as part of our baseline performed tests. We determined the hyperparameter values using the following values:

- Optimizer = Adam
- Learning rate = 0.009 and 0.5 step size step-scheduler
- Window size = 5
- Convolutional kernel size TCN = 3
- Transformer encoders = 2
- Layers of the encoder's hidden units = 1
- Encoders dropout = 0.2

A. DATASET SOURCES

We use nine datasets in our experiments (eight public data sets). Table III shows the details of datasets. As an illustration, the SMAP dataset contains 55 distinct entities, each with 25 dimensions.

1) *Numenta Anomaly Benchmark (NAB)* is an actual data stream containing marked exceptions from various sources, ranging from social media to temperature sensors to server network utilization (Su et al. 2019). We removed incorrectly tagged sequences of anomalies from this dataset for our performed tests.
2) *HexagonML (UCR)* is a multivariate time series dataset used in the KDD 2021 cup (Dau et al. 2019). We only used the portion of the dataset obtained from the real world.
3) *MIT-BIH Supraventricular Arrhythmia Database (MBA)* contains standard test materials for arrhythmia detectors (Moody & Mark 2001). This dataset has been used in around 500 studies of cardiac dynamics.
4) *Soil Moisture Active Passive (SMAP)* is a 25-dimensional dataset collected by NASA that contains telemetry information anomaly data extracted from Anomalous Event Anomaly (ISA) reports from spacecraft monitoring systems (Hundman et al. 2018).
5) *Mars Science Laboratory (MSL)* is a SMAP-like dataset that includes actuator and sensor data from the Mars rover itself. We used only three non-trivial sequences (A4, C2, and T1) dataset in Hundman et al. (2018).
6) *Secure Water Treatment (SWaT)* consists of data obtained from 51 sensors in a continuously operating water treatment system (Mathur & Tippenhauer 2016). The data includes water level, flow rate, and other sensor readings.
7) *Server Machine Dataset (SMD)* was gathered over five weeks from a major internet company (Zhao et al. 2020). SMD was split into two sets of the same size, one used for training and the other for testing. Only the four non-trivial sequences from this dataset were utilized.
8) *Multi-Source Distributed System (MSDS)* consists of application logs, metrics, and distributed traces from a multi-source distributed system (Zhang et al. 2021).
9) *Water Distribution (WADI)* refers to an expansion of the SWaT system, which includes over two times the sensors and actuators compared to the original SWaT model. Additionally, the dataset was obtained over a longer period of time, covering 14 days for normal scenarios and two days for attack scenarios system (Ahmed et al. 2017).

TYPE	DIMENSIONS	TRAIN	VALIDATION	ANOMALIES RATE (%)
MSDS	10 (1)	146430	146430	5.37
SMD	38 (4)	708420	708420	4.16
SWaT	51 (1)	496800	449919	11.98
MSL	55 (3)	58317	73729	10.72
SMAP	25 (55)	135183	427617	13.13
MBA	2 (8)	100000	100000	0.14
UCR	1 (4)	1600	5900	1.88
NAB	1 (6)	4033	4033	0.92
WADI	123 (1)	1048571	172801	5.99

Table III Dataset characteristics.

B. RESULT AND ANALYSIS

We comprehensively compared our newly proposed algorithm, KBJNet, and several state-of-the-art algorithms in the field, such as MSCRED, MAD-GAN, USAD, MTAD-GAT, CAE-M, GDN, and DTAAD. To evaluate the performance of these algorithms, we employed a set of relevant metrics, including Precision (P), Recall (R), Area Under Curve (AUC), and F1 scores. We partition the data into 80% and 20% subsets for training purposes, respectively. This division allows us to examine how the models perform when provided with limited training examples and when trained on a larger volume of data. By assessing the model's behavior in these contrasting scenarios, we can gain valuable insights into its scalability and generalization capabilities and identify potential challenges that may arise in real-world applications with varying data availability. This evaluation provides a comprehensive understanding of how our models perform with substantial data and a limited dataset, allowing us to make informed decisions regarding their suitability for different operational environments.

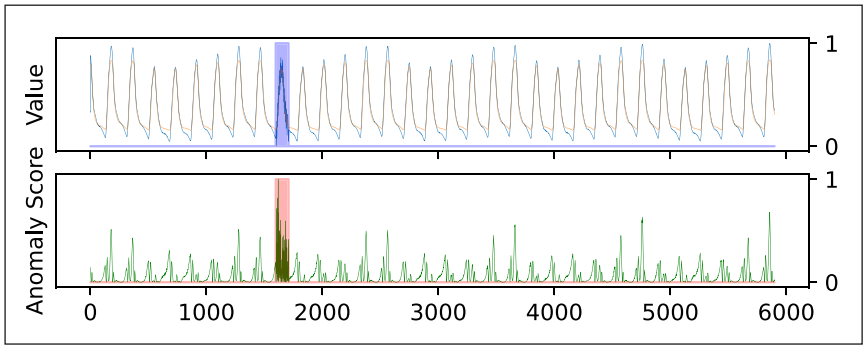

Figure 5 Results in UCR.

1) Performance with 20% of the training dataset: Recently developed models, including unsupervised anomaly detection (USAD), multivariate time-series anomaly detection via graph attention networks (MTAD-GAT), and graph deviation networks (GDN), utilize attention mechanisms to concentrate on particular features of the data and capture long-term trends by adjusting neural network weights. However, KBJNet, which utilizes self-attention, outperforms USAD, MTAD-GAT, and GDN across all datasets as shown Table V. USAD and MTAD-GAT have constraints when classifying anomalies that occur over an extended period because they only consider a local contextual window. To surpass this restriction, KBJNet utilizes self-conditioning on embedding the entire trace along with position encoding, which enhances temporal attention, except for DTAAD on the MBA dataset. The utilization of a meta-learning strategy with MAML enables KBJNet to swiftly acquire anomaly features within sequential data, even with a limited dataset volume (Figure 5). By employing only 20% of the available data, the performance of TranAD and DTAAD closely approaches that of KBJNet, primarily due to their utilization of a generative adversarial training approach for training the encoder-decoder structure. In general, KBJNet demonstrates better performance compared to all other methods.

2) Performance with 80% of the training dataset: Table IV provided illustrates a comparison between the KBJNet approach and other baseline methods in terms of performance metrics related to anomaly detection.

The POT method is used in models such as TranAD, DTAAD, and KBJNet to determine more precise threshold values by considering localized peak values in data sequences. Models like MSCRED use sequential observations as input and retain temporal information, but they may not detect anomalies close to normal trends. KBJNet addresses this issue by amplifying errors using a bi-joint network, enabling it to detect even mild anomalies in datasets such as SMD, where abnormal data is relatively close to regular data, shown in Figure 10.

MSCRED is effective in storing time information due to its continuous observation and good performance on partial datasets, but it struggles to identify anomalies close to normal and operates at a lower speed. The KBJNet architecture can effectively capture information from various dimensions simultaneously. At the same time, KBJNet can efficiently track input and capture long-range dependencies due to Position Encoding and residual connections. As seen

Table IV Comparison of KBJNet model with baseline methods with 80% of the training dataset.

METHOD	NAB				UCR				MBA				SMAP				SWaT			
	P	R	AUC	F1	P	R	AUC	F1	P	R	AUC	F1	P	R	AUC	F1	P	R	AUC	F1
MSCRED	0.8521	0.6700	0.8400	0.7501	0.5440	0.9717	0.9919	0.6975	0.9271	1.0000	0.9798	0.9622	0.8174	0.9215	0.9820	0.8663	0.9991	0.6769	0.8432	0.8071
MAD-GAN	0.8665	0.7011	0.8477	0.7751	0.8537	0.9890	0.9983	0.9164	0.9395	1.0000	0.9835	0.9688	0.8156	0.9215	0.9890	0.8653	0.9592	0.6956	0.8462	0.8064
USAD	0.8421	0.6667	0.8332	0.7443	0.8953	1.0000	0.9990	0.8953	0.8954	0.9990	0.9702	0.9444	0.7481	0.9628	0.9890	0.8419	0.9977	0.6879	0.8460	0.8143
MTAD-GAT	0.8422	0.7273	0.8222	0.7803	0.7813	0.9973	0.9979	0.8762	0.9019	1.0000	0.9720	0.9483	0.7992	0.9992	0.9846	0.8882	0.9719	0.6958	0.8465	0.8110
CAE-M	0.7919	0.8020	0.8020	0.7969	0.6982	1.0000	0.9958	0.8223	0.8443	0.9998	0.9662	0.9155	0.8194	0.9568	0.9902	0.8828	0.9698	0.6958	0.8465	0.8102
GDN	0.8130	0.7873	0.8543	0.7999	0.6895	0.9989	0.9960	0.8159	0.8833	0.9893	0.9529	0.9333	0.7481	0.9892	0.9865	0.8519	0.9698	0.6958	0.8463	0.8102
TranAD	0.8889	0.9892	0.9541	0.9364	0.9407	1.0000	0.9994	0.9694	0.9576	1.0000	0.9886	0.9783	0.8104	0.9998	0.9887	0.8953	0.9977	0.6879	0.8438	0.8143
DTAAD	0.8889	0.9999	0.9996	0.9412	0.8880	1.0000	0.9988	0.9407	0.9608	1.0000	0.9896	0.9800	0.8220	0.9999	0.9911	0.9023	0.9697	0.6957	0.8462	0.8101
KBJNet	0.8889	0.9999	0.9996	0.9412	0.9999	1.0000	**0.9999**	**0.9999**	**0.9805**	1.0000	**0.9898**	**0.9805**	**0.8302**	0.9999	**0.9901**	**0.9072**	**0.9718**	0.6957	**0.8463**	**0.8109**

METHOD	SMD				MSL				MSDS				WADI			
	P	R	AUC	F1	P	R	AUC	F1	P	R	AUC	F1	P	R	AUC	F1
MSCRED	0.7275	0.9973	0.9920	0.8413	0.8911	0.9861	0.9806	0.9362	0.9998	0.7982	0.8942	0.8878	0.2512	0.7318	0.8411	0.3740
MAD-GAN	0.9990	0.8439	0.9932	0.9149	0.8515	0.9929	0.9861	0.9168	0.9981	0.6106	0.8053	0.7578	0.2232	0.9123	0.8025	0.3587
USAD	0.9061	0.9975	0.9934	0.9496	0.7949	0.9912	0.9795	0.8822	0.9913	0.7960	0.8980	0.8829	0.1874	0.8297	0.8724	0.3057
MTAD-GAT	0.8211	0.9216	0.9922	0.8684	0.7918	0.9825	0.9890	0.8769	0.9920	0.7965	0.8983	0.8835	0.2819	0.8013	0.8822	0.4170
CAE-M	0.9081	0.9670	0.9782	0.9368	0.7752	1.0000	0.9904	0.8734	0.9909	0.8440	0.9014	0.9115	0.2783	0.7917	0.8727	0.4118
GDN	0.7171	0.9975	0.9925	0.8343	0.9309	0.9893	0.9815	0.9592	0.9990	0.8027	0.9106	0.8900	0.2913	0.7932	0.8778	0.4261
TranAD	0.9051	0.9973	0.9933	0.9490	0.9037	0.9999	0.9915	0.9493	0.9998	0.8625	0.9012	0.8904	0.3959	0.8295	0.8998	0.5360
DTAAD	0.8463	0.9974	0.9892	0.9147	0.9038	0.9999	**0.9918**	0.9495	**0.9999**	0.8026	0.9013	0.8905	**0.9017**	0.3910	0.6950	0.5455
KBJNet	**0.9985**	0.9974	**0.9987**	**0.9985**	0.9038	0.9999	0.9916	**0.9496**	0.9592	**0.9554**	**0.9248**	**0.9573**	0.8465	**0.8296**	**0.9130**	**0.8379**

Table V Comparison of KBJNet model with baseline methods with 20% of anomalies dataset.

METHOD	NAB		UCR		MBA		SMAP		MSL		SWaT		SMD		MSDS		WADI	
	AUC*	F1*	AUC*	F1*	AUC*	F1*	AUC*	F1*	AUC*	F1*	AUC*	F1*	AUC*	F1*	AUC*	F1*	AUC*	F1*
MSCRED	0.8298	0.7012	0.9636	0.4928	0.9498	0.9107	0.9810	0.8049	0.9796	0.8231	0.8384	0.7921	0.9767	0.8003	0.7715	0.8282	0.6028	0.0412
MAD-GAN	0.8193	0.7108	0.9958	0.8215	0.9549	0.9191	0.9876	0.8467	0.9648	0.8189	0.8455	0.8011	0.8634	0.9317	0.5001	0.7389	0.5382	0.0936
USAD	0.7268	0.6782	0.9968	0.8539	0.9698	0.9426	0.9884	0.8380	0.9650	0.8191	0.8439	0.8088	0.9855	0.9214	0.7614	0.8390	0.7012	0.0734
MTAD-GAT	0.6957	0.7012	0.9975	0.8672	0.9689	0.9426	0.9815	0.8226	0.9783	0.8025	0.8460	0.8080	0.9799	0.6662	0.6123	0.8249	0.6268	0.0521
CAE-M	0.7313	0.7127	0.9927	0.7526	0.9617	0.9003	0.9893	0.8313	0.9837	0.7304	0.8459	0.7842	0.9570	0.9319	0.6002	0.8390	0.6110	0.0782
GDN	0.8300	0.7014	0.9938	0.8030	0.9672	0.9317	0.9888	0.8412	0.9415	0.8960	0.8391	0.8073	0.9812	0.7108	0.6820	0.8390	0.6122	0.0413
TranAD	0.9216	0.8420	0.9983	0.9211	0.9946	0.9897	0.9884	0.8936	0.9856	0.9171	0.8461	0.8093	0.9847	0.8794	0.8112	0.8389	0.6852	0.0698
DTAAD	0.9330	0.9057	0.9984	0.9220	**0.9955**	**0.9912**	0.9894	0.8996	0.9864	0.9212	0.8460	0.8087	0.9866	0.8941	0.8115	0.8390	0.7818	0.0977
KBJNet	**0.9999**	**0.9231**	**0.9999**	**0.9328**	0.9932	0.9869	**0.9894**	**0.9007**	**0.9907**	**0.9451**	0.8460	0.8087	**0.9986**	**0.9983**	**0.9829**	**0.9107**	**0.8453**	**0.1511**

in Figure 8, TranAD, DTAAD, and KBJNet demonstrate advantages over other models because they utilize meta-learning to accelerate model training. Among other models, MSCRED and GRU from the MTAD-GAT model make their operation speed quite inefficient as they are not executed in parallel. On large-volume datasets, their training time is slower than KBJNet. Apart from KBJNet, USAD considers time performance optimization with limited effect. Therefore, USAD and MAD-GAN adopt generative adversarial training, making USAD less computationally intensive than MAD-GAN. Figures 6 and 7 illustrate the training time and inference time in all datasets.

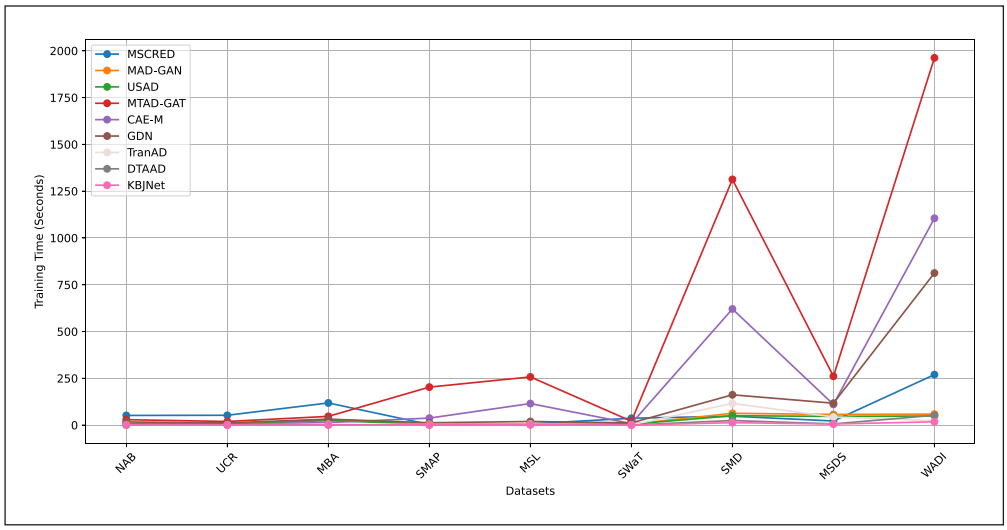

Figure 6 Training time in all datasets.

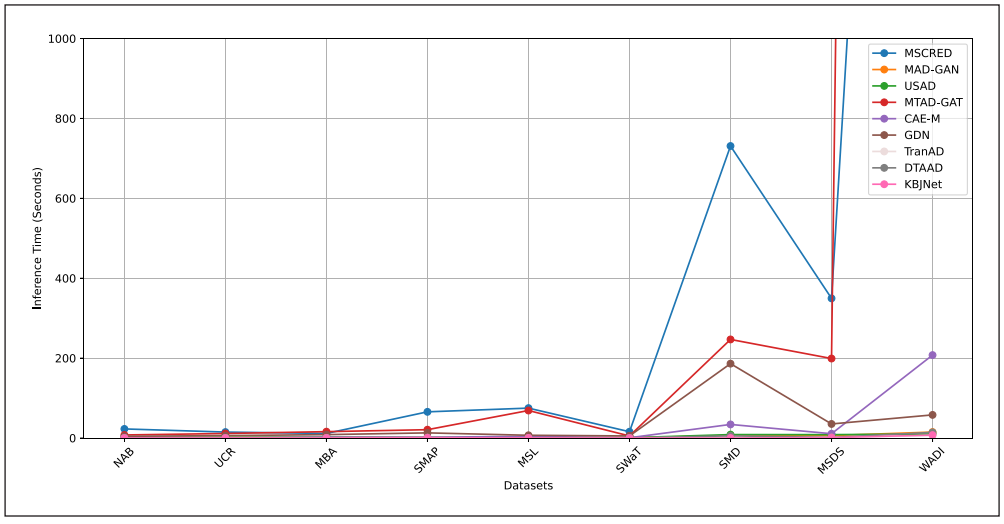

Figure 7 Inference time in all datasets.

3) Sensitivity to the number of training epoch: The correlation between the performance of the anomaly detection model and the number of training epochs is illustrated in Table VI. It reveals that the model's recall rate remains consistently high at 0.9974 across all training epochs. This indicates that the model can accurately identify the significance of the true positive cases and has a low rate of false negatives, which is important for effectively detecting anomalies in non-normal datasets. The AUC score, which evaluates the model's performance, increases from 0.9200 in the first epoch to 0.9985 in the tenth. This indicates that the model's ability to accurately differentiate between anomalies and normal data points improves with increased training epochs. The F1-Score shows an increasing trend from 0.9393 in the second epoch to 0.9972 in the tenth. This suggests that the model achieves a better balance between precision and recall as the number of training epochs increases, which is important for an effective anomaly detection model.

4) Sensitivity to window size: In this study, we present our findings derived from three multivariate datasets: SMD, MSDS, and WADI. This choice is based on the consistently better

EPOCH	PRECISION	RECALL	AUC	F1-SCORE
1	0.9567	0.8440	0.9200	0.8968
2	0.8876	0.9974	0.9922	0.9393
3	0.8831	0.9974	0.9919	0.9368
4	0.8996	0.9974	0.9929	0.9460
5	0.9662	0.9974	0.9969	0.9815
6	0.9985	0.9974	0.9986	0.9979
7	0.9996	0.9974	0.9987	0.9985
8	0.9992	0.9974	0.9986	0.9983
9	0.9985	0.9974	0.9986	0.9979
10	0.9970	0.9974	0.9985	0.9972

Table VI The connection epochs and the performance on SMD datasets.

performance demonstrated by KBJNet across diverse datasets. Increasing the window size can affect the time dependency values in the data. A larger window size will result in increased dependency on other data points. This enhancement also impacts the speed of anomaly detection. Figure 8 illustrates the detection results for four window sizes across three datasets. Better performance is observed with window sizes of 5 and 20 for SMD and 20 for WADI. The results suggest that smaller windows are more suitable for datasets with weak dependencies. In the case of the SMD dataset, a decrease in performance is evident when the window size reduces the model's generalization ability. Moreover, larger windows increase memory and computational requirements, thus slowing down the training process.

5) Sensitivity to MAML: The utilization of MAML enables KBJNet to swiftly discern unusual patterns in sequential data, even when dealing with a limited dataset (Table VII). The response of KBJNet to different datasets with varying K values in a sensitivity analysis is contingent upon the specific dataset under consideration. The effectiveness of MAML varies based on the degree of similarity between the meta-tasks and the target task. The findings suggest that selecting smaller K values in MAML is more suitable. In the case of the MSL dataset, we observe a deterioration in performance as K increases in SMD, impacting both computational efficiency and overall performance. Furthermore, larger K values impose greater computational demands and result in a slowdown of the training process.

6) Sensitivity to kernel size: In these findings, we maintained the global TCN layer and adjusted the filter size by altering the receptive field. Once again, we experimented using SMD, MSDS, and WADI datasets. The results are presented in Figure 9. Optimal performance was achieved for the SMD and MSDS datasets, with a slight decrease observed for WADI. Therefore, kernel size becomes a consideration. However, due to the consistent expansion factor, kernel size changes do not significantly impact the final results.

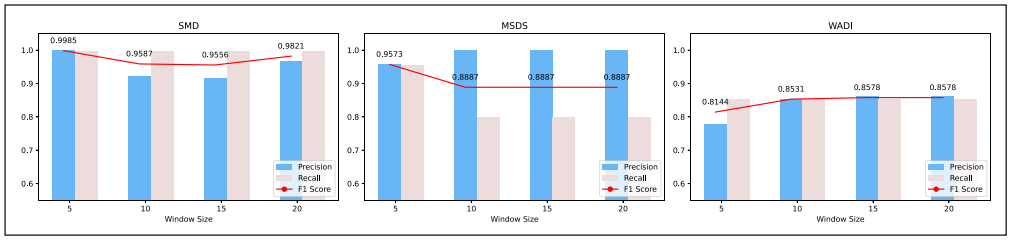

Figure 8 Sensitivity to window size.

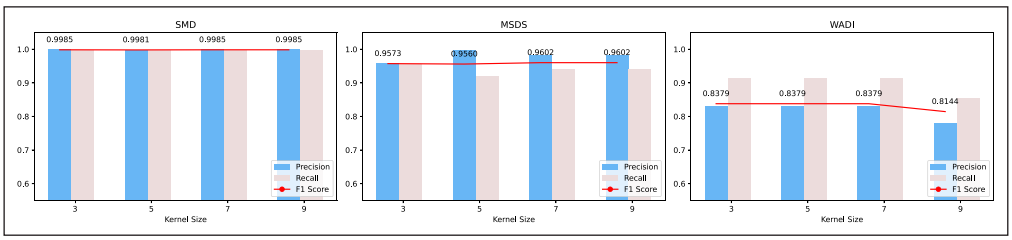

Figure 9 Sensitivity to kernel size.

DOI: 10.5334/dsj-2024-010

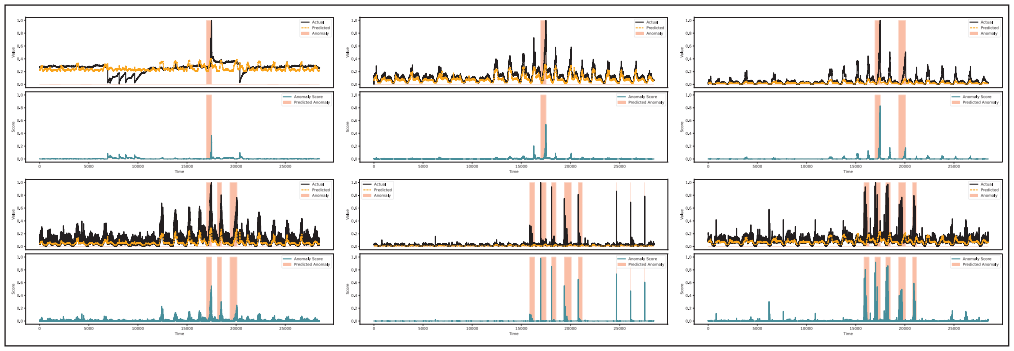

Figure 10 Ground truth and predicted for the SMD using the KBJNet.

7) Ablation analysis: Table VIII summarizes the F1 scores and AUC values for KBJNet and its ablated versions, each with 80% of the training dataset. First, our proposed KBJNet model has proven effective as it achieves the highest performance regarding both AUC and F1 scores on most datasets.

We conducted ablation experiments on the KBJNet model to evaluate the impact of each component by removing the bi-joint TCN, MAML, and transformer from the KBJNet model. From Table VIII, by observing the results, it is evident that eliminating the bi-joint TCN module slightly reduces the F1 scores for most datasets. However, its effect on the AUC scores of the

METHODS	5	10	15	20
NAB	**0.9231**	0.9057	0.9057	**0.9231**
UCR	0.9328	0.9328	0.9328	0.9328
MBA	0.9869	**0.9871**	0.9867	**0.9871**
SMAP	0.9007	0.8926	0.8926	**0.9338**
MSL	**0.9451**	0.8998	0.8998	0.8998
SWaT	0.8087	0.8087	**0.8094**	0.8087
SMD	**0.9983**	0.9970	0.9820	0.9983
MSDS	0.9107	0.9107	0.9107	0.9107
WADI	**0.1511**	0.1104	0.1208	0.1071

Table VII Sensitivity KBJNet to MAML 20% datasets according to meta step-size.

COMPONENT	NAB		UCR		MBA	
	AUC	F1	AUC	F1	AUC	F1
KBJNet	**0.9996**	**0.9412**	**0.9999**	**0.9999**	**0.9898**	**0.9805**
(-)Bi-Joint TCN	0.9996	0.9411	0.9986	0.9327	0.9898	0.9787
(-)MAML	0.9996	0.9412	0.9990	0.9527	0.9889	0.9787
(-)Transformer	0.9325	0.9050	0.9980	0.9188	0.9926	0.9858
COMPONENT	**SMAP**		**MSL**		**SWaT**	
	AUC	F1	AUC	F1	AUC	F1
KBJNet	**0.9901**	**0.9072**	**0.9916**	**0.9496**	**0.8463**	**0.8109**
(-)Bi-Joint TCN	0.9903	0.9083	0.9565	0.7848	0.8462	0.8101
(-)MAML	0.9890	0.8974	0.9573	0.7878	0.8462	0.8101
(-)Transformer	0.9853	0.8682	0.9700	0.8412	0.8459	0.8086
COMPONENT	**SMD**		**MSDS**			
	AUC	F1	AUC	F1		
KBJNet	**0.9987**	**0.9985**	**0.9248**	**0.9573**		
(-)Bi-Joint TCN	0.9911	0.8732	0.9809	0.8991		
(-)MAML	0.9923	0.8790	0.9784	0.8872		
(-)Transformer	0.9852	0.8582	0.9789	0.8937		

Table VIII F1 scores and AUC for KBJNet with 80% of the training datasets.

UCR, MBA, SMAP, and MSL datasets is more pronounced. This indicates that the bi-joint TCN module contributes significantly to capturing temporal dependencies and enhancing the overall effectiveness of the KBJNet model.

Next, we observe that removing the MAML module has a greater impact on the F1 scores than on the AUC values of most datasets, indicating that the MAML module contributes to improving the model's ability to adapt to new tasks and data distributions. Finally, removing the transformer module exerts the greatest influence on the AUC values of the NAB and MSL datasets. This suggests that the transformer module is essential for capturing global contextual information and enhancing the model's discriminative power. Figure 6 reveals that KBJNet requires significantly less time than the baseline methods. These findings indicate the lightweight nature of our model and highlight the benefits of incorporating positional encoding.

In summary The Table VIII, our ablation study confirms that the KBJNet model's component contributes to performance as a whole in anomaly detection, with the bi-joint TCN module playing the most critical role in capturing temporal dependencies, followed by the MAML module for better adaptation to new tasks and the transformer module for capturing global contextual information.

V. CONCLUSION

This research developed the KBJNet, a novel anomaly detection model based on bi-joint TCN, which accurately identifies anomalies within multivariate time series data. Leveraging the power of the transformer architecture, our model adeptly handles lengthy data sequences.

Through rigorous experimentation across nine benchmark datasets, KBJNet outperforms established state-of-the-art methods, yielding substantial enhancements in F1 and F1* scores, ranging from 2% to 9%, for complete and compact datasets, respectively. We noticed that our algorithm did not surpass all aspects of the other algorithms. However, it is worth highlighting that KBJNet exhibited superior performance to most algorithms under consideration. Furthermore, KBJNet is versatile and can adapt for deployment across diverse devices, making it particularly well-suited for contemporary industrial and embedded systems demanding accurate and efficient anomaly detection.

To ensure a more comprehensive assessment of its efficacy, further experimentation with datasets from diverse fields will be beneficial. This broader testing approach will enable us to determine the model's applicability and performance in various contexts beyond the industrial domain. Optimizing our model's efficiency remains open to further research, potentially enhancing processing speed and resource utilization.

FUNDING INFORMATION

This research is funded by The Department of Informatics, Institut Teknologi Sepuluh Nopember.

COMPETING INTERESTS

The authors have no competing interests to declare.

AUTHOR AFFILIATIONS

Muhammad Abdan Mulia
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Muhammad Bintang Bahy
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Muhammad Zain Fawwaz Nuruddin Siswantoro
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Nur Rahmat Dwi Riyanto
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Nella Rosa Sudianjaya
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

Ary Mazharuddin Shiddiqi orcid.org/0000-0002-8762-3141
Department of Informatics, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

REFERENCES

Ahmed, CM, Palleti, VR and **Mathur, AP.** 2017. WADI. In: *Proceedings of the 3rd International Workshop on Cyber-Physical Systems for Smart Water Networks*, ACM. DOI: https://doi.org/10.1145/3055366.3055375

Amer, M and **Goldstein, M.** 2012. Nearest-neighbor and clustering based anomaly detection algorithms for rapidminer. In: *Proceedings of the 3rd RapidMiner Community Meeting and Conference (RCOMM 2012)*, pp. 1–12.

Bandaragoda, TR, Ting, KM, Albrecht, D, Liu, FT and **Wells, JR.** 2014. Efficient anomaly detection by isolation using nearest neighbour ensemble. In: *2014 IEEE International Conference on Data Mining Workshop*, IEEE. DOI: https://doi.org/10.1109/ICDMW.2014.70

Boniol, P, Palpanas, T, Meftah, M and **Remy, E.** 2020. GraphAn. *Proceedings of the VLDB Endowment*, 13(12): 2941–2944. DOI: https://doi.org/10.14778/3415478.3415514

Breunig, MM, Kriegel, H-P, Ng, RT and **Sander, J.** 2000. LOF. In: *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, ACM. DOI: https://doi.org/10.1145/342009.335388

Budiarto, EH, Permanasari, AE and **Fauziati, S.** 2019. Unsupervised anomaly detection using k-means, local outlier factor and one class SVM. In: *2019 5th International Conference on Science and Technology (ICST)*, IEEE. DOI: https://doi.org/10.1109/ICST47872.2019.9166366

Bulusu, S, Kailkhura, B, Li, B, Varshney, PK and **Song, D.** 2020. Anomalous example detection in deep learning: A survey. *IEEE Access*, 8: 132330–132347. DOI: https://doi.org/10.1109/ACCESS.2020.3010274

Chatterjee, A and **Ahmed, BS.** 2022. IoT anomaly detection methods and applications: A survey. *Internet of Things*, 19: 100568. DOI: https://doi.org/10.1016/j.iot.2022.100568

Chen, J, Mao, Q and **Liu, D.** 2020. Dual-path transformer network: Direct context-aware modeling for end-to-end monaural speech separation. In: *Interspeech 2020*, ISCA. DOI: https://doi.org/10.21437/Interspeech.2020-2205

Chen, Z, Chen, D, Zhang, X, Yuan, Z and **Cheng, X.** 2022. Learning graph structures with transformer for multivariate time-series anomaly detection in IoT. *IEEE Internet of Things Journal*, 9(12): 9179–9189. DOI: https://doi.org/10.1109/JIOT.2021.3100509

Chowdhury, S, Deb, A, Nurujjaman, M and **Barman, C.** 2017. Identification of pre-seismic anomalies of soil radon-222 signal using Hilbert–Huang transform. *Natural Hazards*, 87(3): 1587–1606. DOI: https://doi.org/10.1007/s11069-017-2835-1

Darban, ZZ, Webb, GI, Pan, S, Aggarwal, CC and **Salehi, M.** 2022. Deep learning for time series anomaly detection: A survey. arXiv:2211.05244.

Dau, HA, Bagnall, A, Kamgar, K, Yeh, C-CM, Zhu, Y, Gharghabi, S, Ratanamahatana, CA and **Keogh, E.** 2019. The UCR time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6): 1293–1305. DOI: https://doi.org/10.1109/JAS.2019.1911747

Deng, A and **Hooi, B.** 2021. Graph neural networkbased anomaly detection in multivariate time series. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5): 4027–4035. DOI: https://doi.org/10.1609/aaai.v35i5.16523

Dhiman, H, Deb, D, Muyeen, SM and **Kamwa, I.** 2021. Wind turbine gearbox anomaly detection based on adaptive threshold and twin support vector machines. *IEEE Transactions on Energy Conversion*, 36(4): 3462–3469. DOI: https://doi.org/10.1109/TEC.2021.3075897

Ergen, T and **Kozat, SS.** 2020. Unsupervised anomaly detection with LSTM neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 31(8): 3127–3141. DOI: https://doi.org/10.1109/TNNLS.2019.2935975

Finn, C, Abbeel, P and **Levine, S.** 2017. Model-agnostic meta-learning for fast adaptation of deep networks, In: *International conference on machine learning*, PMLR, pp. 1126–1135.

Goldstein, M and **Dengel, A.** 2012. Histogram-based outlier score (hbos): A fast unsupervised anomaly detection algorithm, *KI-2012: poster and demo track*, 1: 59–63.

Hu, W, Gao, J, Li, B, Wu, O, Du, J and **Maybank, S.** 2020. Anomaly detection using local kernel density estimation and context-based regression. *IEEE Transactions on Knowledge and Data Engineering*, 32(2): 218–233. DOI: https://doi.org/10.1109/TKDE.2018.2882404

Huang, S, Liu, Y, Fung, C, He, R, Zhao, Y, Yang, H and **Luan, Z.** 2020. HitAnomaly: Hierarchical transformers for anomaly detection in system log. *IEEE Transactions on Network and Service Management*, 17(4): 2064–2076. DOI: https://doi.org/10.1109/TNSM.2020.3034647

Hundman, K, Constantinou, V, Laporte, C, Colwell, I and **Soderstrom, T.** 2018. 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. DOI: https://doi.org/10.1145/3219819.3219845

Jin, Y, Qiu, C, Sun, L, Peng, X and **Zhou, J.** 2017. Anomaly detection in time series via robust PCA. In: *2017 2nd IEEE International Conference on Intelligent Transportation Engineering (ICITE)*, IEEE, pp. 352–355. DOI: https://doi.org/10.1109/ICITE.2017.8056937

Kingsbury, K and **Alvaro, P.** 2020. Elle. *Proceedings of the VLDB Endowment*, 14(3): 268–280. DOI: https://doi.org/10.14778/3430915.3430918

Kiss, I, Genge, B, Haller, P and **Sebestyen, G.** 2014. Data clustering-based anomaly detection in industrial control systems. In: *2014 IEEE 10th International Conference on Intelligent Computer Communication and Processing (ICCP)*, IEEE. DOI: https://doi.org/10.1109/ICCP.2014.6937009

Li, D, Chen, D, Jin, B, Shi, L, Goh, J and **Ng, S-K.** 2019. Mad-GAN: Multivariate anomaly detection for time series data with generative adversarial networks. In: *Artificial Neural Networks and Machine Learning–ICANN 2019: Text and Time Series: 28th International Conference on Artificial Neural Networks*, Munich, Germany, September 17–19, 2019, Proceedings, Part IV, Springer, pp. 703–716. DOI: https://doi.org/10.1007/978-3-030-30490-4_56

Li, Z, Zhao, Y, Botta, N, Ionescu, C and **Hu, X.** 2020. COPOD: Copula-based outlier detection. In: *2020 IEEE International Conference on Data Mining (ICDM)*, IEEE. DOI: https://doi.org/10.1109/ICDM50108.2020.00135

Luo, Y, Chen, Z and **Yoshioka, T.** 2020. Dual-path RNN: Efficient long sequence modeling for time-domain single-channel speech separation. In: *ICASSP 2020–2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE. DOI: https://doi.org/10.1109/ICASSP40776.2020.9054266

Malhotra, P, Ramakrishnan, A, Anand, G, Vig, L, Agarwal, P and **Shroff, G.** 2016. Lstm-based encoder-decoder for multi-sensor anomaly detection. arXiv preprint arXiv:1607.00148.

Mathur, AP and **Tippenhauer, NO.** 2016. SWaT: A water treatment testbed for research and training on ICS security. In: *2016 International Workshop on Cyber-physical Systems for Smart Water Networks (CySWater)*, IEEE. DOI: https://doi.org/10.1109/CySWater.2016.7469060

Moody, G and **Mark, R.** 2001. The impact of the MIT-BIH arrhythmia database. *IEEE Engineering in Medicine and Biology Magazine*, 20(3): 45–50. DOI: https://doi.org/10.1109/51.932724

Pan, X, Tan, J, Kavulya, S, Gandhi, R and **Narasimhan, P.** 2010. Ganesha, *ACM SIGMETRICS Performance Evaluation Review*, 37(3): 8–13. DOI: https://doi.org/10.1145/1710115.1710118

Park, D, Hoshi, Y and **Kemp, CC.** 2018. A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. *IEEE Robotics and Automation Letters*, 3(3): 1544–1551. DOI: https://doi.org/10.1109/LRA.2018.2801475

Patcha, A and **Park, J-M.** 2007. An overview of anomaly detection techniques: Existing solutions and latest technological trends. *Computer Networks*, 51(12): 3448–3470. DOI: https://doi.org/10.1016/j.comnet.2007.02.001

Provotar, OI, Linder, YM and **Veres, MM.** 2019. Unsupervised anomaly detection in time series using LSTM-based autoencoders. In: *2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT)*, IEEE. DOI: https://doi.org/10.1109/ATIT49449.2019.9030505

Qu, Z, Su, L, Wang, X, Zheng, S, Song, X and **Song, X.** 2018. A unsupervised learning method of anomaly detection using GRU. In: *2018 IEEE International Conference on Big Data and Smart Computing (BigComp)*, IEEE. DOI: https://doi.org/10.1109/BigComp.2018.00126

Ramaswamy, S, Rastogi, R and **Shim, K.** 2000. Efficient algorithms for mining outliers from large data sets. *ACM SIGMOD Record*, 29(2): 427–438. DOI: https://doi.org/10.1145/335191.335437

Salem, O, Guerassimov, A, Mehaoua, A, Marcus, A and **Furht, B.** 2014. Anomaly detection in medical wireless sensor networks using SVM and linear regression models. *International Journal of E-Health and Medical Communications*, 5(1): 20–45. DOI: https://doi.org/10.4018/ijehmc.2014010102

Sarker, IH. 2021. Data science and analytics: An overview from data-driven smart computing, decision-making and applications perspective. *SN Computer Science*, 2(5). DOI: https://doi.org/10.1007/s42979-021-00765-8

Schölkopf, B, Platt, JC, Shawe-Taylor, J, Smola, AJ and **Williamson, RC.** 2001. Estimating the support of a highdimensional distribution. *Neural Computation*, 13(7): 1443–1471. DOI: https://doi.org/10.1162/089976601750264965

Shang, W, Cui, J, Song, C, Zhao, J and **Zeng, P.** 2018. Research on industrial control anomaly detection based on FCM and SVM. In: *2018 17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications/12th IEEE International Conference on Big Data Science and Engineering (TrustCom/BigDataSE)*, IEEE. DOI: https://doi.org/10.1109/TrustCom/BigDataSE.2018.00042

Shih, S-Y, Sun, F-K and **yi Lee, H.** 2019. Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 108(8–9): 1421–1441. DOI: https://doi.org/10.1007/s10994-019-05815-0

Shyu, M-L, Chen, S-C, Sarinnapakorn, K and **Chang, L.** 2003. A novel anomaly detection scheme based on principal component classifier. Technical Report, Miami Univ Coral Gables Fl Dept of Electrical and Computer Engineering.

Siffer, A, Fouque, P-A, Termier, A and **Largouet, C.** 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. DOI: https://doi.org/10.1145/3097983.3098144

Su, Y, Zhao, Y, Niu, C, Liu, R, Sun, W and **Pei, D.** 2019. 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*, ACM. DOI: https://doi.org/10.1145/3292500.3330672

Tax, DM and **Duin, RP.** 2004. Support vector data description. *Machine Learning*, 54: 45–66. DOI: https://doi.org/10.1023/B:MACH.0000008084.60811.49

Thill, M, Konen, W, Wang, H and **Bäck, T.** 2021. Temporal convolutional autoencoder for unsupervised anomaly detection in time series. *Applied Soft Computing*, 112: 107751. DOI: https://doi.org/10.1016/j.asoc.2021.107751

Thudumu, S, Branch, P, Jin, J and **Singh, JJ.** 2020. A comprehensive survey of anomaly detection techniques for high dimensional big data. *Journal of Big Data*, 7(1). DOI: https://doi.org/10.1186/s40537-020-00320-x

Tran, L, Mun, MY and **Shahabi, C.** 2020. Real-time distance-based outlier detection in data streams. *Proceedings of the VLDB Endowment*, 14(2): 141–153. DOI: https://doi.org/10.14778/3425879.3425885

Tuli, S, Casale, G and **Jennings, NR.** 2022. TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data. *Proceedings of VLDB*, 15(6): 1201–1214. DOI: https://doi.org/10.14778/3514061.3514067

Veličković, P, Cucurull, G, Casanova, A, Romero, A, Liò, P and **Bengio, Y.** 2018. Graph Attention Networks. *International Conference on Learning Representations*. Available at: https://openreview.net/forum?id=rJXMpikCZ

Wang, B, Hua, Q, Zhang, H, Tan, X, Nan, Y, Chen, R and **Shu, X.** 2022. Research on anomaly detection and real-time reliability evaluation with the log of cloud platform. *Alexandria Engineering Journal*, 61(9): 7183–7193. DOI: https://doi.org/10.1016/j.aej.2021.12.061

Wang, Y, Masoud, N and **Khojandi, A.** 2021. Real-time sensor anomaly detection and recovery in connected automated vehicle sensors. *IEEE Transactions on Intelligent Transportation Systems*, 22(3): 1411–1421. DOI: https://doi.org/10.1109/TITS.2020.2970295

Xu, J, Wu, H, Wang, J and **Long, M.** 2022. Anomaly transformer: Time series anomaly detection with association discrepancy. In: *International Conference on Learning Representations*. Available at: https://openreview.net/forum?id=LzQQ89U1qm

Yaacob, AH, Tan, IK, Chien, SF and **Tan, HK.** 2010. ARIMA based network anomaly detection. In: *2010 Second International Conference on Communication Software and Networks*, IEEE. DOI: https://doi.org/10.1109/ICCSN.2010.55

Yin, K, Yang, Y, Yao, C and **Yang, J.** 2022. Long-term prediction of network security situation through the use of the transformer-based model. *IEEE Access*, 10: 56145–56157. DOI: https://doi.org/10.1109/ACCESS.2022.3175516

Yu, L, Lu, Q and **Xue, Y.** 2023. DTAAD: Dual TCN-attention networks for anomaly detection in multivariate time series data. arXiv:2302.10753. DOI: https://doi.org/10.2139/ssrn.4410420

Zang, D, Liu, J and **Wang, H.** 2018. Markov chain-based feature extraction for anomaly detection in time series and its industrial application. In: *2018 Chinese Control And Decision Conference (CCDC)*, IEEE, pp. 1059–1063. DOI: https://doi.org/10.1109/CCDC.2018.8407286

Zhang, C, Song, D, Chen, Y, Feng, X, Lumezanu, C, Cheng, W, Ni, J, Zong, B, Chen, H and **Chawla, NV.** 2019. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(1): 1409–1416. DOI: https://doi.org/10.1609/aaai.v33i01.33011409

Zhang, Y, Chen, Y, Wang, J and **Pan, Z.** 2021. Unsupervised deep anomaly detection for multi-sensor timeseries signals, *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1. DOI: https://doi.org/10.1109/TKDE.2021.3102110

Zhao, H, Wang, Y, Duan, J, Huang, C, Cao, D, Tong, Y, Xu, B, Bai, J, Tong, J and **Zhang, Q.** 2020. Multivariate time-series anomaly detection via graph attention network. In: *2020 IEEE International Conference on Data Mining (ICDM)*, IEEE. DOI: https://doi.org/10.1109/ICDM50108.2020.00093

Zong, B, Song, Q, Min, MR, Cheng, W, Lumezanu, C, Cho, D and **Chen, H.** 2018. Deep autoencoding gaussian mixture model for unsupervised anomaly detection. In: *International Conference on Learning Representations*.

TO CITE THIS ARTICLE:
Abdan Mulia, M, Bahy, MB, Siswantoro, MZFN, Riyanto, NRD, Sudianjaya, NR and Shiddiqi, AM. 2024. KBJNet: Kinematic Bi-Joint Temporal Convolutional Network Attention for Anomaly Detection in Multivariate Time Series Data. *Data Science Journal*, 23: 10, pp. 1–22. DOI: https://doi.org/10.5334/dsj-2024-010

Submitted: 29 June 2023
Accepted: 06 October 2023
Published: 04 March 2024

Data Science Journal is a peer-reviewed open access journal published by Ubiquity Press.