International Journal of Engineering Trends and Technology
ISSN: 2231–5381 / https://doi.org/10.14445/22315381/IJETT-V72I9P123

Volume 72 Issue 9, 283-296, September 2024

Original Article

Unlocking Long-Term Temporal Patterns: TCAE for Anomaly Detection in Multivariate Time Series Data

Sangeeta oswal[1], Subhash Shinde[2], M. Vijayalakshmi[3]

[1,2]*Lokmanya Tilak College of Engineering, Maharashtra, Navi Mumbai, India.*
[3]*Vivekanand Education Society's Institute of Technology, Maharashtra, Mumbai, India.*

[1]*Corresponding Author : sangita.oswal14@gmail.com*

Received: 04 May 2024 Revised: 08 August 2024 Accepted: 26 August 2024 Published: 28 September 2024

Abstract - *Anomaly detection in multivariate time series is extremely important in modern industrial systems to mitigate attacks and minimize system downtime. Anomalies are often seen as subtle deviations from established normal patterns. The challenge of learning extended temporal patterns in time series remains unresolved, hindering effective anomaly detection. To address the above challenge in this study, a novel approach, termed Temporal Convolutional Auto Encoder (TCAE), is introduced. TCAE utilizes Temporal Convolution Networks (TCN) and employs casual convolutions and dilations to effectively simulate long-term dependency in sequential data, taking advantage of its temporality and large fields. The autoencoder is trained on normal operations to learn the temporal dependencies present in the input time series. Two anomaly detection strategies employing the Local Outlier Factor (LOF) and thresholding are investigated. A supervised grid search technique is employed to determine the threshold, optimizing the model's performance. The thresholding technique demonstrates a performance improvement of over 20% when compared to the average performance of other baseline models.*

Keywords - *Anomaly detection, Auto encoder, Deep learning, Local Outlier Factor, Multivariant Time Series, Temporal Convolution Network.*

1. Introduction

Anomaly detection in time series entails identifying patterns in the data that significantly depart from the anticipated normal behaviour. The application of identifying anomalous behaviour is growing in fraud detection, networking, predictive maintenance, and health monitoring systems [1]. The computing and communications infrastructure makes modern Industrial Control Systems (ICS) a primary target for cyber-attacks due to their increased connectedness to the internet. Sophisticated IT technologies are needed to manage physical processes in Critical Infrastructures (CIs) such as power grids, water treatment plants, etc. [2]. The majority of this collection consists of time series data. An essential aspect of ensuring successful service quality control in manufacturing industries, simulation processes, test beds, and cyber-physical systems is the detection of anomalies in the huge amount of data they generate. Nevertheless, the intricate temporal interdependence of multivariate time series makes anomaly detection a significant hurdle. Attacks on ICS have consequences for the environment and safety; hence, developing and deploying an anomaly detector system to mitigate this attack is important. Deep learning has demonstrated an exceptional aptitude for acquiring intricate datasets, including temporal and high-dimensional data, thereby pushing the boundaries of many learning tasks [3]. Its practical applications in engineering problem-solving have generated significant research interest. Unsupervised anomaly detection is preferred over a supervised setup because of the scarcity of labels [4].

Most anomaly detection algorithms operate under the assumption that a significant portion of data instances exhibit normal behaviour and may be learned. During deployment, the model can leverage its understanding of common or expected patterns to distinguish between regular and irregular (anomalous) patterns [5]. Anomaly detection task is challenged by complex long-range temporal dependency. Recurrent neural networks cannot effectively model complex time co-relation because it requires processing step-by-step recursion. The majority of deep learning models that employ sequential networks for time series processing are not able to capture long-term dependencies in time series [6-8]. Furthermore, networks such as LSTM and RNNs are limited by their sequential processing nature, where each subsequent step relies on the processing of all preceding stages, leading to longer inference time due to a low degree of parallelism [9]. A model that can capture high-level temporal dependencies with minimal overhead is needed. To address the above challenge, this study presents a novel Temporal Convolution Auto Encoder architecture that is referred to as TCAE and utilizes

convolutional neural networks as its foundation. The proposed model utilizes a TCN-inspired autoencoder to represent normal time stamps and applies it to detect abnormal patterns that depart from the expected behavior. When dealing with time series data, the TCAE excels at handling information over lengthy periods. The TCAE algorithm utilizes dilated convolutional layers to efficiently learn time series with long and intricate temporal patterns. The dilated convolution employs a wide receptive field and analyzes the information at multiple temporal scales. The TCAE model trains encoders and decoders simultaneously. Encoders learn to compress input time series while decoders recreate them. The reconstruction error is used to detect abnormal behavior. The rationale behind this technique is that the architectural bottleneck compels the network to detect valuable temporal patterns in the data, hence facilitating efficient representation of the input. The paper's main contributions are as follows:

1. Parallelism: TCN can process multiple sequences simultaneously, unlike RNN, which processes them sequentially.
2. Adaptable perceptual scope: fine-tuning the TCN model with hyperparameters, including the number of layers, convolutional kernel size, and expansion coefficient, which define the size of TCN's perceptual field.
3. Optimal Threshold: Evaluating Local Outlier Factor (LOF) and thresholding techniques to identify anomalous data points. The thresholding technique conducts a tradeoff analysis between precision and recall to determine the appropriate threshold.
4. Data-Centric Design: Examine the level of design knowledge required for creating an anomaly detector that makes use of a data-centric method.

TCNs like CNN are less prone to the issue of exploding or vanishing gradients that occur in sequence learning problems using RNN family networks. The rest of the paper is organized as follows: Section 2 presents the related work. Section 3 is the methodology used in the research work. Section 4 represents the data set used and highlights the result achieved, followed by a discussion in Section 5. Lastly, the conclusion and future work is presented.

2. Literature Review

While numerous unsupervised approaches using ML and statistical methods exist, they are unable to handle the correlation; hence, it is harder to effectively capture the robust representation considering the temporal dependency present in multivariate time series [11-13]. Deep learning methods exhibit superior performance compared to traditional approaches when it comes to modeling the dependency in multivariant time series. Consequently, a significant obstacle in predicting multivariate time series is determining how to accurately represent the changing relationship between time steps to model long-term dependency across numerous variables. Most of the deep learning solutions for modeling time series for anomaly detection are either prediction or reconstruction-based, leveraging recurrent networks (RNN, LSTM, etc.), GANs, autoencoders, or variational autoencoders. However, the general problems with this sequence modeling technique, pioneered by RNN, struggle to capture distant associations due to the gradient vanishing issue. LSTM and GRU-based models are proposed to mitigate the issue with RNN. Chung et al. [14] experimented with sequential data using GRU and LSTM. Saad et al. [15] worked on seven types of DL models using LSTM and GRU for tacking imputation in time series.

Malhotra et al. [16] developed stacked LSTM models for fault detection in time series. The proposed model is trained on non-anomalous data, and the prediction error is modeled as a multivariant Gaussian distribution, which is used to assess the likelihood of anomalous behavior. The model used an ECG dataset, which necessitates significant computational resources and exhibits very poor processing speeds for high-dimension datasets. Ji et al. [17] proposed an LSTM-based anomaly detection method for univariate time series. The author also used the publicly available ECG dataset for experimental evaluation.

Additionally, they are unable to consistently and efficiently mimic long-term patterns. DAGMM [7] employs an autoencoder based on LSTM and utilizes the reconstruction error to identify anomalous data points through the Gaussian mixture model. Park et al. [18] employ a fusion of LSTM-VAE to address the challenging problem of high modality. The proposed model projects each time it steps into a latent space using serially connected LSTM and VAE layers and uses a dynamic threshold that changes over the estimated state.

Su et al. [19] employ GRU to learn the robust latent representation. The proposed model omnianomaly extends the research to interpret the anomalies. The threshold selection is done using the POT algorithm. Chen et al. [20] utilize an adversarial autoencoder called DAEMON. The model regularizes the hidden variables and reconstructs the data using the adversarial generation method. The top k dimension with the largest reconstruction error is also presented in the study.

Several empirical studies have focused on Generative adversarial networks utilizing LSTM and sequential networks for time series anomaly detection. Gieger et al. [21] utilize LSTM in the generator of GAN and use two critics for the discriminator; for anomaly detection, reconstruction loss is used. The overfitting of reconstruction loss challenges the model. Li et al. [22] proposed MAD-GAN to represent the dependencies across multiple sensors and actuators in the system for detecting anomalies. MAD-GAN used LSTM and RNN neural networks for generators, and discriminator and reconstruction loss were used to identify anomalous data points. Basar et al. [23] employed LSTM in the generator model with three stacked layers with 32, 64, and 128 hidden

units and used single-layer LSTM in the discriminator. Across all these studies, there is consistent evidence to suggest that GAN-based models are sensitive to the number of epochs and are not suitable to fully exploit the spatial-temporal correlation across multiple variables; model stability in the training process is also an issue with GAN[10].

Many studies have focused on Graph neural networks to represent the time series relation across multiple variables. Chen et al. [24] proposed a framework based on graphs. The framework detects anomalies based on the deviation between the inter- and intra-series patterns from anomalous to normal. Deng et al. [25] propose a graph deviation network to model the relationship between sensors using a graph network and detect deviation from this pattern. Ge et al. [26] designed an autoregressive task to model temporal dimension and a graph to model spatial dimension.

Additionally, some researchers applied the Graph Attention Network (GAT). Zhao et al. [27] employ each feature of a multivariate time series as univariate and process it with two GAT network feature-oriented and time-oriented GAT called MTAD-GAT. The model uses both reconstruction and prediction loss for inference. Zhou et al. [28] also employ two GATs simultaneously, but in contrast to MTAD-GAT, GAN is used for reconstruction loss and MLP for prediction loss.

The relation between sensor and actuator is represented by Zhao et al. [29], which calculates a feature matrix and extracts its feature using a convolution encoder and time dimension using a convLSTM unit. The feature matrix is reconstructed using a convolution decoder and uses a threshold to determine anomalies. Zhang[9] proposes MSCRED, which first constructs the signature matrices. Subsequently, a convolutional encoder encodes the inter-sensor (time series) correlations, and an attention-based Convolutional Long-Short Term Memory (ConvLSTM) network captures the temporal patterns.

In addition to the model mentioned above, Thill et al. [30] proposed TCN-AE for the ECG dataset. The paper addresses the challenge of the ECG dataset, where anomalies are the same peak as normal time stamps but in different shapes. He et al. [31] also employ TCN -AE on the ECG dataset; however, the prediction error is fitted to a multivariate Gaussian distribution to calculate the anomaly score. On the other hand, some researchers utilize the transformer. Tuli et al. [32] propose TranAD, which employs a transformer in an adversarial setup for anomaly detection. Kim et al.[33] utilizes multiple transformer encoders and a decoder layer. The decoder layer includes 1D convolution to fuse the representation of multiple encoders. It also uses thresholds to detect anomalies. Yu et al. [34] employ a transformer utilizing TCN for anomaly detection. Zeng et al.[35] also proposed an adversarial transformer with fused probability for anomaly detection. In the proposed model, the anomaly score is based on reconstruction error plus anomaly probability, which determines the probability of the current time stamp being anomalous.

For threshold selection, the previous study LSTM-NDT [8] used nonparametric thresholding techniques, which demonstrated poor results. The POT [36] approach for thresholding is employed in models like USAD[6], MTAD-GAT[27], TranAD [32], and DTAAD [34]. The acronym POT, which originates from Peaks Over the Threshold, represents the second principle of Extreme Value Theory (EVT). Instead of using human threshold setting and distribution assumptions, the POT method uses "extreme value theory", which relies on the Generalized Pareto Distribution (GPD) to examine data and choose the suitable value-at-risk (label) for dynamically establishing the threshold.The POT technique finds and extracts the highest values over a threshold. Use a generalized Pareto distribution to examine and model the data. The small application range of the POT technique makes threshold selection the most difficult [37]. The approach recommended by the proposed model uses reconstruction loss distribution to generate a threshold that maximizes performance measures.

The key findings of the Literature review are summarized:

- Sequential modeling using LSTM's GRU RNN, etc., suffers from slow convergence due to recurrent connection and is not effective in capturing long-term dependency.
- GAN-based models are sensitive, and the stability of the GAN model in high-dimension data is a challenge.
- The local contextual window limits graph-based networks to create a Graph for modeling temporal and spatial dependencies.
- Transformers are utilized in several studies [38] and are still in the inspection stage.
- Hybrid models are showing promising results but are challenged by the integration of various techniques to fully exploit the spatial-temporal correlation and other interconnection amongst the multiple variables (sensors/actuators) in the system for detecting anomalies.

To conclude, the recursive model necessitates the transmission of information about all preceding units. Convolutional networks in numerous sequence modeling tasks surpass RNNs and address the common drawbacks of recursive models, such as slow modeling or issues with gradient explosion/disappearance. Also, convolutional networks enable parallel calculation of the output. Temporal Convolutional Networks(TCN), which are dilated convolutional networks [39], can operate in the time domain.

This paper introduces an autoencoder model that utilizes Temporal Convolutional Networks (TCN) and is trained on normal data from the SWaT water treatment plant. After being

trained with data that precisely reflects the regular working of the plant, the autoencoder has the potential to duplicate the typical behaviour of the sensors and actuators in its output. This is the essential notion that underpins the autoencoder. The reconstruction loss is computed for test data that includes both normal and anomalous time series. The underlying notion is that, given that the model is trained on regular operations, the reconstruction loss will be significantly lower than that of anomalous time stamps. Two approaches are compared to flag anomaly. First, LOF is used to model abnormal behaviour and is contrasted to the thresholding technique. Thresholding labels abnormal behaviour when reconstruction loss exceeds the threshold; otherwise, it is normal.

3. Methods

Given a training input time-series T, our goal is to forecast $Y = \{y_1, y_2, \ldots . y_t\}$ for any test time series $\hat{t}$ that has the same characteristics as the training time series. Here, $y_t \in \{0, 1\}$ represents whether the data point at the t^{-th} timestamp of the test data is anomalous (1 signifies an anomalous point).

3.1. Data Preprocessing

To enhance the effectiveness of the model training, data standardization is employed for both the training and test data segments. The labels are removed for unsupervised processing, and the columns are converted to float and normalize using the Min-Max Scaler.

$$x_t = \frac{x^t - min(T)}{max(T) - min(T) + \epsilon'} \quad (1)$$

Here, min(T) represents the dimension-wise minimum, and max(T) represents the dimension-wise maximum vectors in the training time series. A very small constant ϵ is added to avoid division by zero. A local contextual window of length 12 is taken, and the time series is converted to sliding window $W = \{W_1, W_2, ..., W_t\}$. The total training window size is (494988, 12, 51) and the test window size is (449907, 12, 51). Here, 12 represent window size, and 51 represent the dimension of time series. In this study, the complete SWaT dataset is taken, unlike other models, which downsamples the data and then evaluates the performance measure.

3.2. TCN

Temporal Convolutional Networks (TCNs) are commonly employed in computer vision tasks, leveraging the advantages of convolutional operations in the temporal domain. The current study focuses on the versatile applications of temporal autoencoders, particularly in anomaly detection within time series data. The TCN is characterized by three key parameters: the kernel size (k), a list of dilation rates ($q_1, q_2, ..., q_L$),) and the number of filters $n_{filters}$. The Temporal Convolutional Autoencoder (TCAE) compresses sequences temporally to generate a compact sequence and subsequently reconstruct the original sequence. Unlike traditional autoencoders, TCAE substitutes dense layers with convolutional architecture, resulting in a reduced number of weights compared to dense autoencoders. The encoder incorporates a down-sampling layer, while the decoder utilizes an up-sampling layer. To maintain consistency in input and output durations, the TCN relies on information from prior time steps, employing zero padding to ensure the output tensor matches the length of the input tensor. The proposed TCAE architecture comprises various building blocks, which will be detailed in the subsequent sections.

3.2.1. Dilation Convolution

Causal convolution is computationally expensive for long sequences. However, dilation convolution increases the magnitude of the receptive field without adding much computational cost. Convolutional layers in neural networks typically handle multivariate time series x[n] of dimension d, with $X: \rightarrow \mathbb{R}^d$. The convolution operation performed between each dimension $x_i[n]$ and the filter $h_j[n]$ resulting in the output y[n]:

$$y[n] = (x * h)[n] = \sum_{i=0}^{k-1} h[i]^t * x[n-i] \quad (2)$$

$y[n] \in R$ is the output, k is filter, and h[i] is the i^{th} weight. During the convolution procedure, the input sequence x[n] is passed through a k-length window with filter weights h[i]. At each time step, the weights are used to compute a weighted average. The filter simply slides along the time axis; hence, the procedure is called one-dimensional convolution. The central idea is to learn suitable weights for the filter based on the task. The dilatation convolution refers to an extra parameter of the dilation rate, which defines the number of elements that are skipped from input signals between filters. The dilation convolution is written as:

$$y[n] = (x * q^h)[n] = \sum_{i=0}^{k-1} h[i]^t * [n - qi] \quad (3)$$

If q =1, it is the same as equation 2. The basic principle is a growing dilation rate throughout a series of dilated convolutional layers. The standard practice is to set the initial network layer's dilation rate at q= 1 and then increase it by a factor of 2 for each subsequent layer. Using this method, the model's receptive field can be expanded at an exponential rate.

$$r_{acausal} = \left[k/2\right](2^{L+1} - 2) + 1 \quad (4)$$

L define the number of layers and K is kernel size. Figure 1 demonstrates the dilation convolution with the dilation factor increasing in the power of 2. This method is employed in time series modeling as it allows for the learning of long-term temporal patterns through the use of vast receptive fields. By utilizing this technique, the model's receptive field is increased without sacrificing resolution, unlike pooling or stride convolutions. Figure 1 presents the dilation convolution where the dilation rate is increased in the power of 2, and the output highlighted in blue depends on all the units in the input.

3.2.2. Temporal Convolution Network Architecture

In essence, a TCN can be described as a series of residual blocks. Figure 2 shows 2 sub-blocks containing a weight normalization layer [22] to normalize the input of hidden, a ReLu activation function after the convolution layer, and a spatial dropout layer to prevent overfitting. Additionally, a skip connection [23] is employed to directly pass through the residual block and add to its output.

The key parameters of the Temporal Convolutional Network are a sequence of dilations (q_1, q_2, …, q_L), the size of the kernel (k), and the number of filters (nfilters).

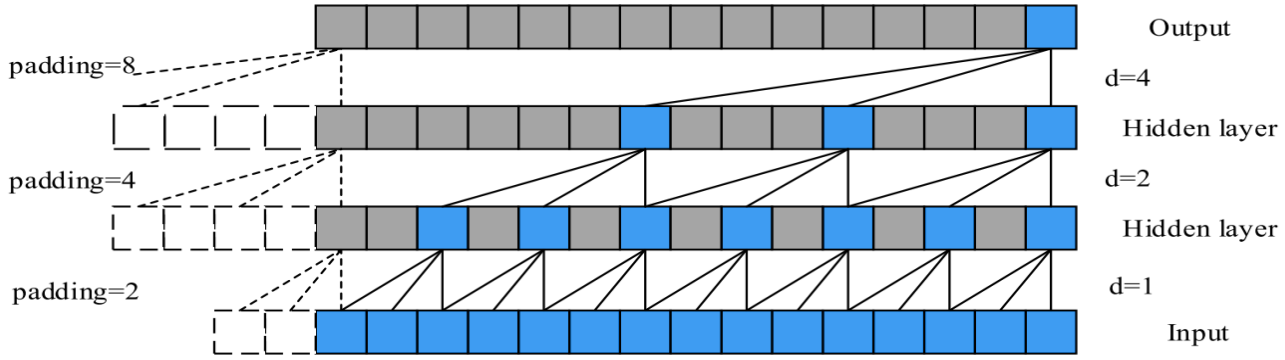

Fig. 1 The dilation convolution

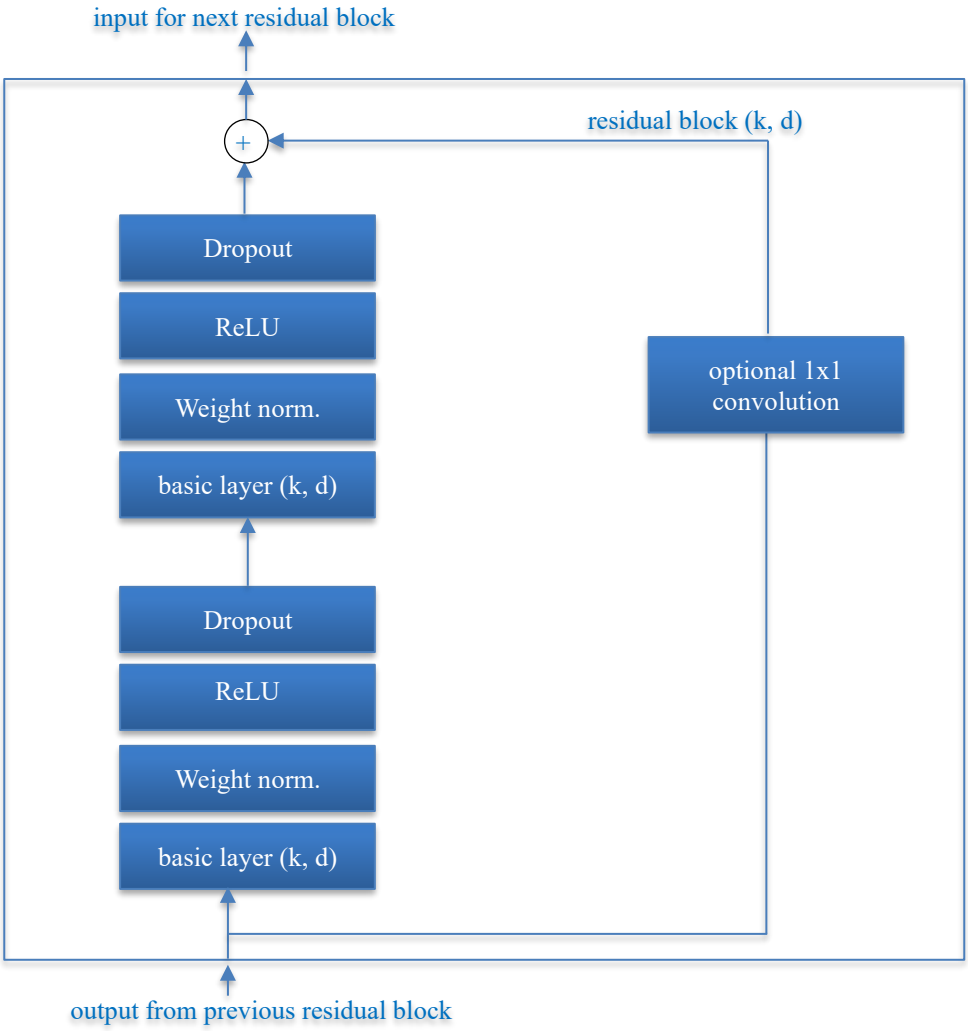

Fig. 2 The TCN network architecture

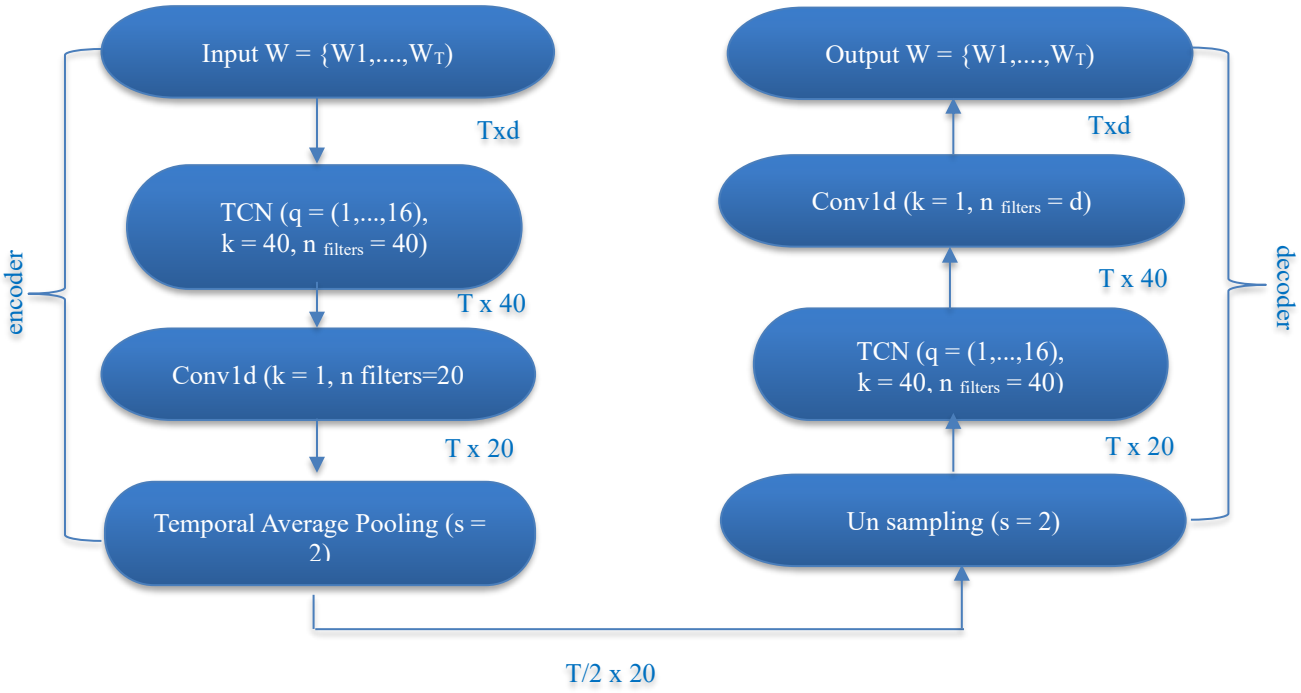

Fig. 3 The TCN AE Model

3.3. Proposed Model: TCAE Model

The encoders strive to create a condensed representation of the input sequence to capture significant features that identify both long-term and short-term dependencies. An autoencoder leveraging Temporal Convolutional Networks (TCN) as a fundamental component is introduced and referred to as a Temporal Convolutional Auto Encoder (TCAE), as illustrated in Figure 3.

The encoder starts by analysing the input sequence x[n] with dimensions d set at 51. The TCN block consists of dilation (q = 1,…,16) increased by a factor of 2. A one-dimensional convolutional layer (1 x 1 convolution) [40] is then used with parameters q = 1, k = 1, and 20 filters to effectively reduce the dimensionality of the feature map (TCN output). The series is down-sampled using an average-pooling layer. Consequently, the original input x[n] undergoes compression, resulting in an encoded representation, where $g : \{0, 1, \ldots, T/s - 1\} \rightarrow R^c$

The decoder module uses an up-sampling layer to perform nearest neighbour interpolation, restoring the sequence to its original length. An additional TCN is used to process the up-sampled sequence; this TCN has independent weights but shares all of the encoder-TCN's parameters. At last, the input sequence is rebuilt using a Conv1D layer, with k = 1 and $n_{filters}$ = d adjusted to match its dimensionality. $\hat{x}[n] = dec(g[n])$, $\hat{x} : T \rightarrow \mathbb{R}^d$. In the subsequent part, the utilization of the input sequence and reconstruction error for the purpose of identifying anomalies is discussed. Figure 4 depicts the training and validation accuracy achieved till epoch 5.

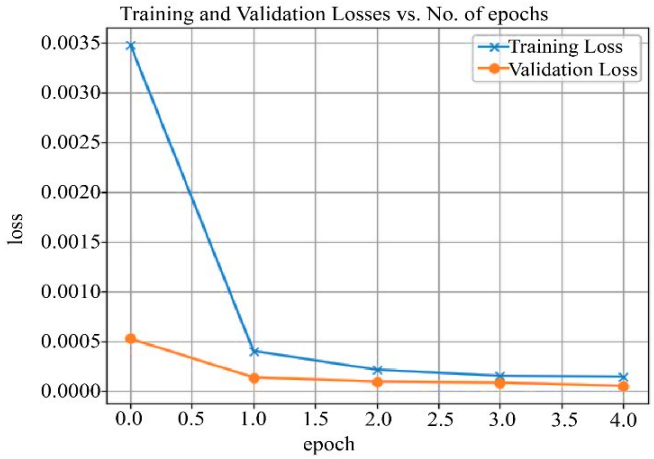

Fig. 4 The Validation and Training loss vs epoch

3.4. Anomaly Detection

TCAE must learn compressed encodings of input sequences for precise reconstruction during training to circumvent the architecture constraint. The TCAE is trained on only normal time series in the training phase with the intuition that the TCAE should recreate normal data points in time series with minimal reconstruction loss. When TCAE detects patterns significantly different from the norm, we anticipate increased reconstruction loss.

For anomaly detection, two methods are evaluated and compared

- Using thresholding technique
- Using LOF without thresholding

Algorithm 1: The TCAE model Training and Anomaly Detection

Initialization : $n_{epochs} = 5, X_{train} = Training\ data, \tau = Anomaly\ Threshold, \tilde{X}[n] = Test\ data$

1. Preprocess data and convert to sliding window
2. Initialize the trainable parameters and construct model TCAE()
3. $for\{1 \ldots\ldots n_{epochs}\}\ do$
4. $Train(TCAE, X_{train})$
5. end for
6. Print the Training and Validation loss and save the model
7. $\hat{X}[n] = TCAE(\tilde{X}[n])$ ▸ encode the test data on saved model
8. E[n] = $\tilde{X}[n]$ - $\hat{X}[n]$ ▸ calculate the reconstruction loss
9. $a(n) = \begin{cases} 1, & e[n] > \tau \\ 0, & else \end{cases}$
10. Return a(n) ▸ for each time series point

3.5. Thresholding Technique

Once the reconstruction loss is obtained for the test data to flag anomalous time series, a thresholding logic is employed; if the reconstruction loss is above a threshold τ we flag the time series as anomalous (1) or else normal (0). The key logic is to find the threshold for which we used 10% of the test data labels with reconstruction loss and perform a grid search to find the optimal threshold, which maximizes the F1 score. Algorithm 1 describes the Temporal Convolutional Auto-Encoder (TCAE) model utilized for anomaly identification. Firstly, the model is trained on a normal dataset during steps 1 to 6 of the algorithms. The saved model is called to predict the reconstruction loss on test data, which contains both normal and anomalous data points in steps 7 and 8. If the reconstruction loss is above the threshold, the data point will be flagged as anomalous.

3.6. LOF without Thresholding

Reconstruction loss is the residue between the predicted value and ground truth values. The Experiment was conducted using the widely used outlier detection method known as Local Outlier Factor (LOF). LOF calculates the local density deviation of a data point with its neighbors. Points with a low LOF are considered anomalous. The selection of the anomaly data point is based on the local neighborhood. The threshold value is not used. We applied the reconstruction loss to LOF to identify anomalous (1) or normal (1) data points in a time series. In the LOF setup, an 11% contamination rate is applied, and the number of neighbors is set to 50. However, it is seen that the thresholding strategies yielded superior results compared to LOF.

4. Results and Discussion

TCAE was trained, validated, and tested on a machine equipped with a GPU A100 and 64GB of RAM. The normal operation records from the complete SWaT dataset are utilized to train and validate the autoencoder by fine-tuning the hyperparameters, as elaborated in section 3.3. The TCN layer employs 40 kernels of size 40, with dilation growing exponentially from 1 to 16, followed by a 1D convolution with 20 filters. Downsampling occurs at a rate of 2. The decoder network shares the same parameters and relearns the weights. The learning rate is 0.001, the kernels are started with glorot normal, the loss function is a mean square error, and the number of epochs is 5.

4.1. Dataset

The SWaT water treatment plant replicates advanced large-scale water treatment systems capable of producing 5 gallons per minute of double-filtered water. It paves the way for academics to verify the efficacy of cyber defenses and examine the response of an operating ICS to cyberattacks. In SWaT, the water treatment process is comprised of six different and interconnected sub-processes, which are denoted as phases P1 - P6. SWaT stages are equipped with sensors and actuators, including water level sensors, control valves, flow meters, and pumps.

They keep tabs on the water's chemical and physical properties at that stage and relay that data to the PLCs. Each Programmable Logic Controller (PLC) will manage operations through an independent network. An 11-day timeframe was used to gather the SWaT dataset, with the system running continuously at 24 hours per day. Over the final four days of the 2016 SWaT data collection phase, a monumental 36 attacks were launched [41]. Generally, the attacked points include sensors (e.g., water level sensors, flow-rate meter, etc.) and actuators (e.g., valves, pumps, etc.). The test bed was subjected to various attacks with distinct objectives and varying durations during the last four attack days.

4.2. Attacks on the SWaT Dataset

Researchers conducted experiments on the SWaT systems to examine cyber-attacks and analyse system responses. A total of 36 attacks were placed into SWaT [42]. For illustrative purposes, an exemplary attack scenario is provided. The objective was to compromise the performance of SWaT from its nominal level, such as 5 gallons per minute, through a targeted attack. The sensor LIT401, which is responsible for sensing the water level of the RO feed tank p4, was compromised in this particular case. The attacker

successfully manipulated LIT401 to decrease the level of the RO feed tank from 800mm to 200mm. This action led to the cessation of pumping by PLC-4 for P401, resulting in a reduction in water flow to P5. The impact of the attack on sensor LIT401 had a detrimental effect on the output water flow rate of the RO unit, as reflected in the FIT501 readings in P5.

To maintain system parameters, the flow rate must adhere to a specific value, approximately 1.2cm/hr., translating to roughly 5 gallons per minute of treated water. The monitored data during the experimental period revealed a decrease in the amount of treated water, underscoring the efficacy of the attack in disrupting the intended system performance.

4.3. Design Consideration

During the first stage of the experiment, two variations were tested to assess how the design of SWaT affects the ability of TCAE to detect anomalies.

4.3.1. Design Centric

Based on the SWaT design, use stage-wise grouping. Thus, six separate detectors monitor each SWaT stage in real time. Each level of SWaT has interdependent water filtration processes.

4.3.2. Data Centric

The autoencoder model operates independently and does not require the design information for Anomaly Detection. While the design-centric only sees a fraction of the plant state, in data-centric, the TCAE performs real-time monitoring of all observable conditions of the entire plant, i.e. all six stages are processed as one unit by the model.

The SWaT architecture features stage dependency, where PLCs in stages are interconnected to ensure the precise operation of sensors and actuators. Data-centric strategies in plant analysis leverage many interdependencies to view all stages as a unified entity. Therefore, only a data-centric approach is employed. To the best of our knowledge, the proposed model TCAE is the first one to consider a complete SWaT dataset for training without downsampling the samples. Table 1 shows the SWaT dataset's information. The entire 496800 data points are preprocessed according to the guidelines outlined in section 3.1. The preprocessed data points are sampled using sliding window and total training data (494988,12,51) consists of 494988 windows where 12 is window size and 51 is the dimension of the SWaT dataset.

Table 1. The Dataset information

Dataset	SWaT
Variables	51
Attacks	38
Training data	496800
Testing data	449919
Anomaly %	11.98%

4.4. Performance Measure

Anomaly detection methods aim to identify anomalies within a defined time series window. To categorize each point as either normal or anomalous, it is necessary to establish an anomaly threshold. Nevertheless, the threshold involves a tradeoff between the number of missed positive cases (recall) and the number of incorrect positive cases (precision). One performance indicator that is utilize is the F1 Score, which compares algorithms based on their precision and recall, ensuring that these two objectives are about equal. A grid search is performed to identify the threshold value that optimizes the F1 score. The concept involves utilizing supervised learning to choose the best threshold from a limited set of time series data with labels and then implementing it over the entire series. A 10% subset is extracted from a time series and analyzed to determine the threshold that produces the highest F1 score. This assessment is repeated ten times, each time utilizing a distinct 10% subset of the data to account for variations. We modify the criterion for the chosen segment and analyze the remaining 90% of the data to determine the average outcomes.

$$precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} \tag{7}$$

4.5. Experimental Results

In this section, the results are presented. The trained model of TCAE is evaluated on test data, which contains 449919 data points. Figure 5 displays the distribution of reconstruction loss for test data with both anomalous and normal timestamps, while Figure 6 shows the box plot of the reconstruction loss.

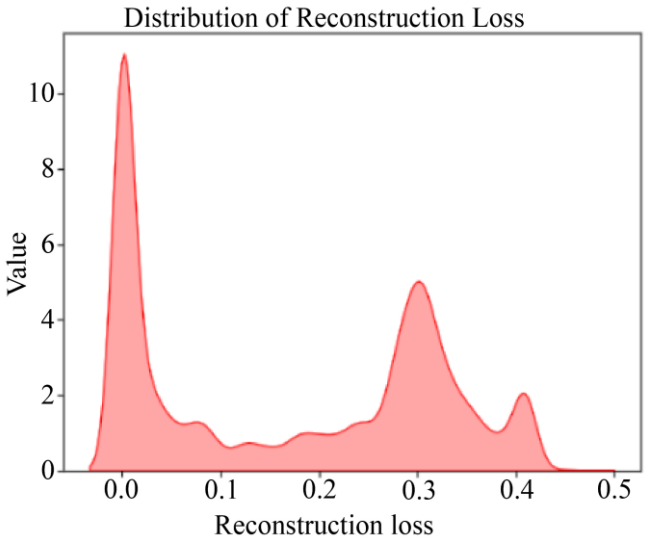

Fig. 5 Distribution of Reconstruction Loss

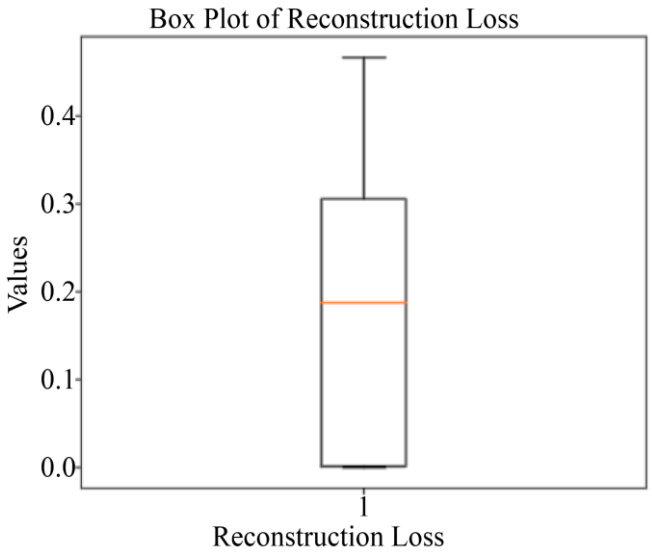

Fig. 6 The box plot of reconstruction loss

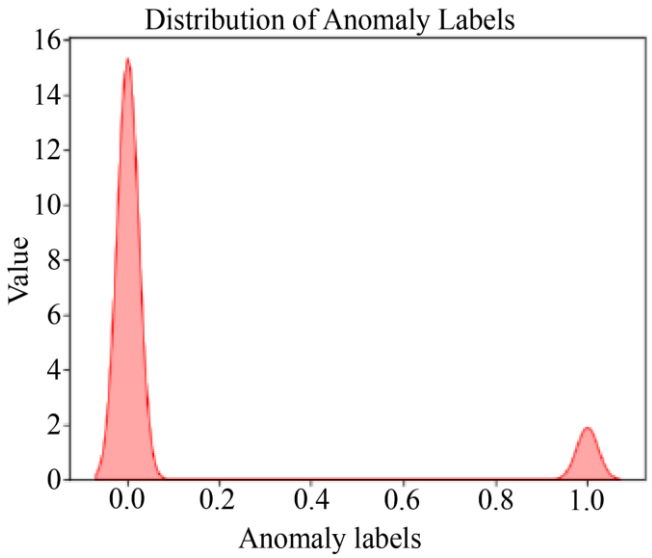

Fig. 7 Distribution of anomaly labels using LOF

The reconstruction loss is processed to identify anomalies. Results for both techniques:

- Thresholding approach and
- LOF

is presented. The LOF approach predicts anomaly labels based on reconstruction loss, with hyperparameters set at 50 neighbors and 11% contamination. Figure 7 shows the distribution of anomaly labels. Label 1 represents anomalies, and label 0 represents regular instances. LOF uses the density of data points in the reconstruction loss as a key factor to detect anomalies. LOF processing is based on the rationale that anomaly data points come from low-density areas and will have higher LOFs. The achieved result is unsatisfactory, with an F1 score of 0.2084 and AUC of 0.57, as shown in Figure 8.

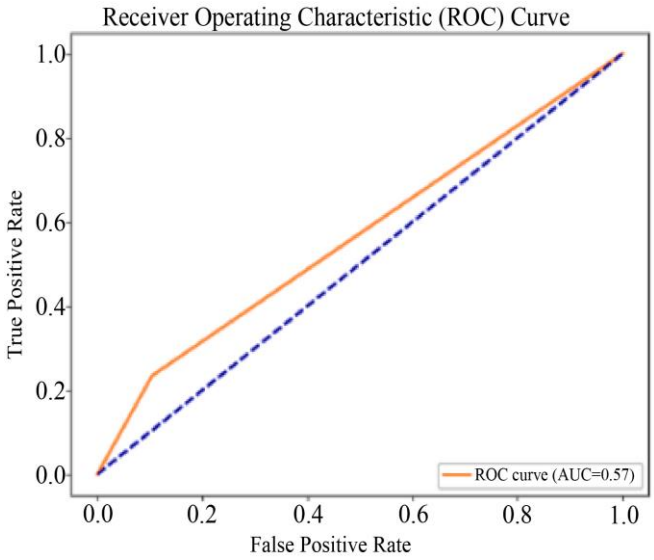

Fig. 8 The ROC curve using LOF

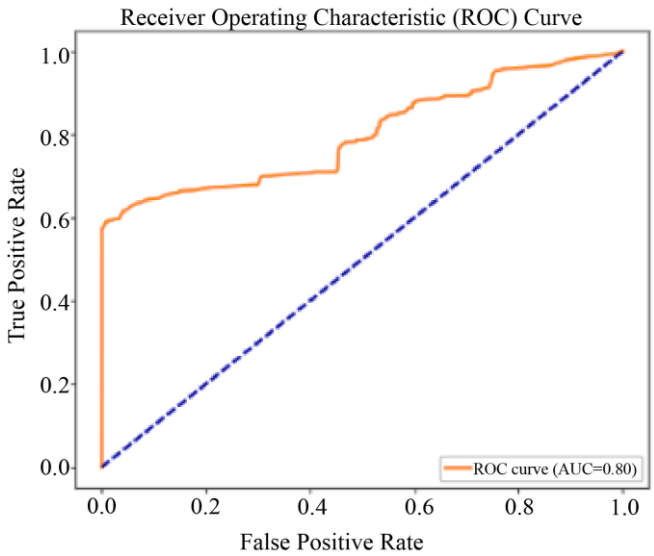

Fig. 9 The ROC Curve using thresholding

In the thresholding technique, the key is to identify the threshold, which is elaborated in section 3.4. If the reconstruction loss is above the identified threshold, the data point is flagged anomalies; otherwise, it is normal. The outcome obtained using the thresholding technique is remarkable. The optimal threshold determined using grid search is 0.3680, resulting in an optimal F1-score of 0.7436. Figure 9 displays the ROC curve with an AUC value of .80 using the thresholding procedure.

Table 2. Results of TCAE vs other models

Model	Precision	Recall	F1 Score
DAGMM	0.4695	**0.6659**	0.5507
USAD	0.7488	0.5945	0.6627
LSTM-NDT	0.7777	0.5108	0.6166
TCAE (Our Model)	**0.9435**	0.6136	**0.7436**

In Table 2, the result is presented and compared with other models, namely USAD, DAGMM and LSTM-NDT. The model USAD, LSTM-NDT, and DAGMM are implemented on the SWaT dataset. The DAGMM model utilizes an autoencoder and Gaussian Mixture Model (GMM) estimator. The USAD utilizes an adversarial-trained autoencoder. LSTM-NDT utilizes LSTM networks and introduces a nonparametric thresholding technique for anomaly detection. The proposed model is compared with three baseline methods that utilize autoencoder and Recurrent Neural Network (RNN) to capture temporal dependencies in time series. The methods are contrasted with our proposed model, which employs a Temporal Convolutional Network (TCN) to model time series. The TCN model demonstrates a strong benefit over traditional dense neural networks by utilizing dilation convolution with a higher receptive field, which proves beneficial for modeling long-term temporal dependencies in time series data. The incorporation of residual connections in TCNs further enhances the stability and efficiency of the learning process. The parallelism makes TCNs faster and more efficient in training and inference, especially on long sequences.

4.6. Discussion

The research focuses on identifying anomalies in multivariate time series data. The Secure Water Treatment (SWaT) testbed on the industrial control system is used to identify anomalies, which are unbalanced data sets with an anomaly rate of 12%. The thresholding technique is optimized for the tradeoff between precision and recall in a high-dimension SWaT dataset (51 dimensions). The efficiency of the proposed model TCAE against the SOTA model is shown in Table 2. It shows that the proposed model has a higher F1 score than the other models. Since the TCAE model uses dilation convolution, it can handle the long-term temporal pattern effectively. Stable gradient and parallelism are other notable advantages of the proposed model TCAE. Furthermore, the dense neural networks are replaced by convolution operations, resulting in computational efficiency for large data sets like SWaT. The proposed model is based on the rationale that anomalous data points will have high reconstruction loss because the model is trained to learn the representation of normal data points.

The key findings of the research works are:

- DGMMM struggles with high-dimension data due to the absence of explicit temporal mapping in the approach and the utilization of a singular GRU model.
- USAD faces challenges in classifying long-term anomalies due to the limited contextual window used for processing.
- The proposed model TCAE learns the robust representation of time series and handles the complex temporal dependency for anomaly detection.
- The research explored the techniques used to decide the threshold. Finding thresholds is challenging because finding accurate boundaries for normal and abnormal data is difficult, as different sensors collect the data.
- The proposed model utilizes TCN, which supports parallel processing and addresses the challenge of sequential networks using LSTM and RNN to model time series.

The graph is presented on the test data, which includes the anomalies in the dataset. The anomalies are predicted by the TCAE model using both LOF and thresholding techniques. Figure 10 represents the plotting of predicted anomaly labels using the LOF method. The Local Outlier Factor (LOF) algorithm is an unsupervised method for detecting anomalies by measuring the deviation in the density of a certain data point to its nearby points. LOF identifies outliers as samples with significantly lower density compared to their neighbors. Figure 11 displays a graph showing the predicted anomaly plotted against the ground truth (actual anomalies) on the reconstruction loss. The blue color symbolizes reconstruction loss. The red dot represents anomalous data points, whereas the yellow dots reflect predicted anomalies. The dashed horizontal line represents a threshold.

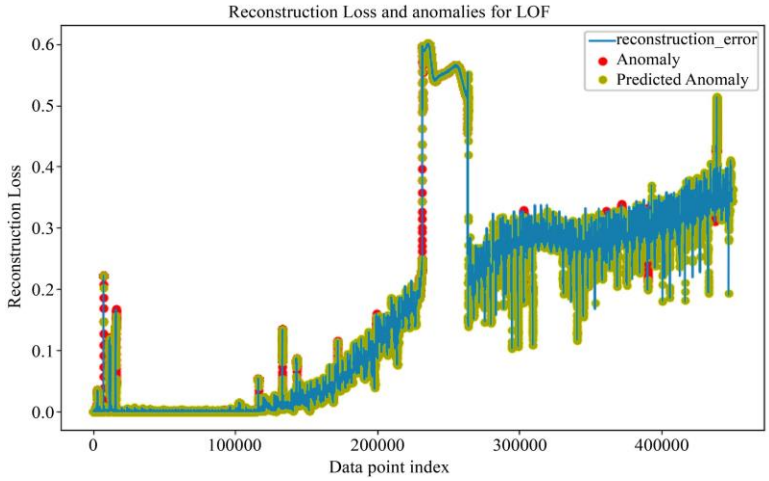

Fig. 10 The Plotting of Reconstruction loss with Predicted anomaly and actual anomaly data point using LOF method without threshold

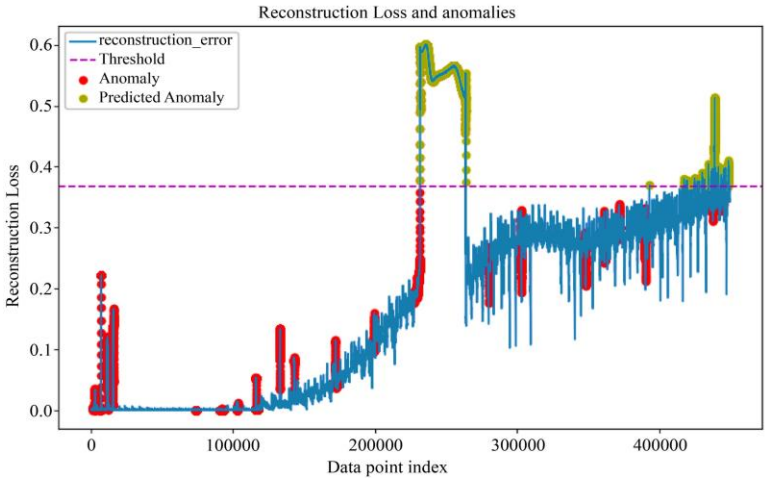

Fig. 11 The Plotting of Reconstruction loss with the Predicted anomaly, actual anomaly data point and threshold value. The data points above the threshold are flagged anomalous, shown in yellow colour

The relationship between threshold values and Precision, Recall, and F1 score is analyzed to determine the best threshold, which falls within the range of 0.36 to 0.38. Figure 12 shows the graph of the F1 score, Precision, and Recall plotted against the Threshold. The tradeoff between precision and recall is adjusted using the F1 score to minimize false positives and balance precision and recall for best performance.

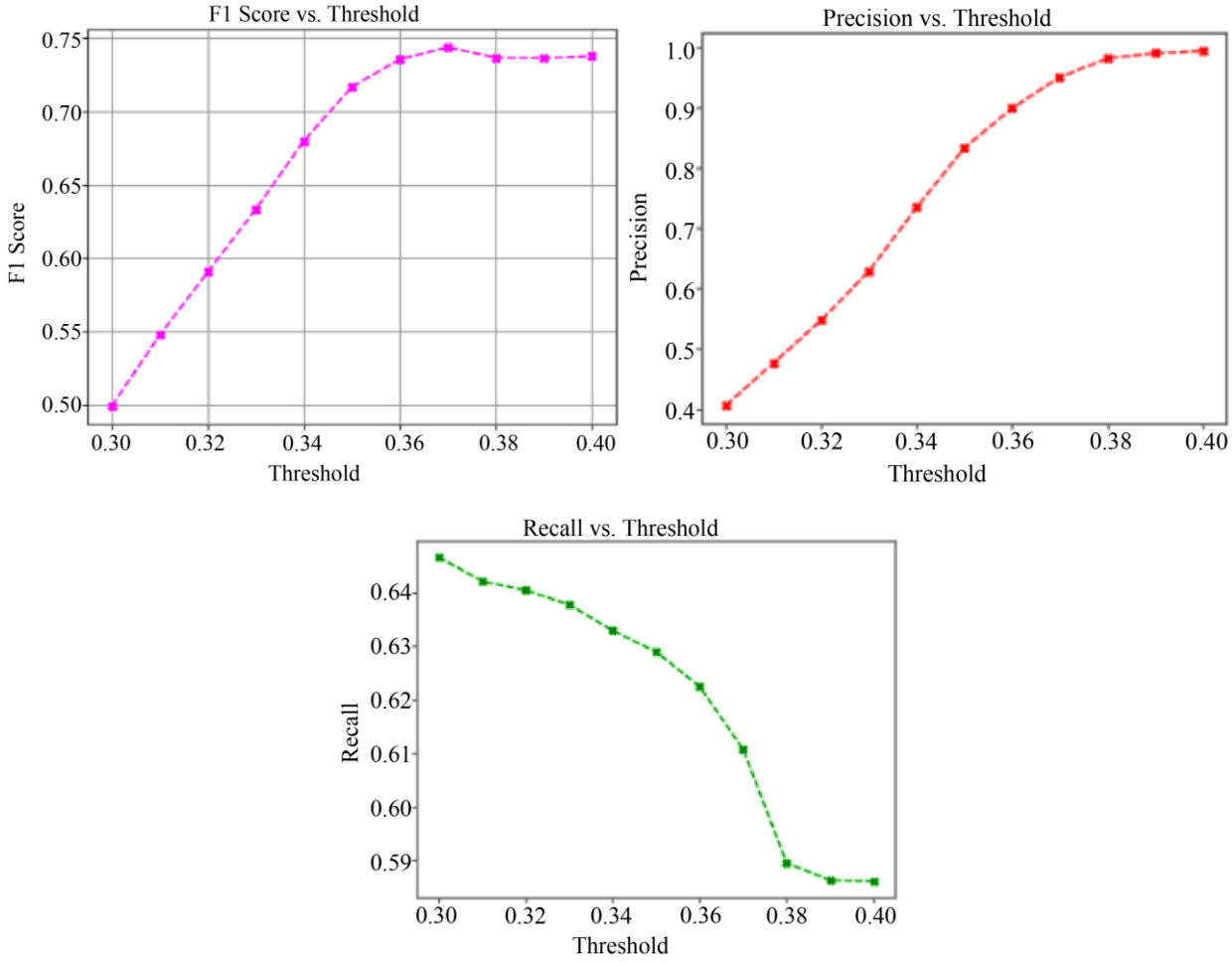

Fig. 12 F1 socre vs threshold, precision vs threshold, recall vs threshold

5. Conclusion and Future Work

This paper presents a TCAE structure that is trained in an unsupervised manner to acquire compact representations of time series data. To the best of our knowledge, this study is the first to showcase the combination of the Temporal Convolutional Network (TCN) and autoencoder (AE) on the SWaT dataset.This study implemented an autoencoder utilizing a temporal convolution network named (TCAE). The TCAE leverages dilated convolutions to enhance the receptive field for effective modeling of long-term dependencies in time series. Two strategies suggested for identifying abnormal data points are the LOF and thresholding approach. The results demonstrate superior performance with the thresholding strategy. The study also evaluates the performance in comparison to various thresholding methods, such as nonparametric thresholding utilized in LSTM-NDT and the POT approach utilize in USAD. To model interdependence between stages, this study employs a data-centric method that treats the entire SWaT dataset as a single unit for processing rather than treating each stage independently. The F1 score shows an overall performance gain of more than 20% compared to the average baseline performance of other models. The TCAE model presented in this paper appears to be well-equipped for capturing complex long-range temporal patterns.

In our upcoming study, we aim to explore uncharted parts of TCAE, such as implementing stacked dilated convolution layers and incorporating a concatenation layer to gather past outputs of each dilated convolution layer. Increasing user confidence by utilizing explainable AI to illustrate the characteristics that lead to abnormal data values is also a further research direction to consider.

Data Availability Statements

SWaT dataset to be obtained from iTrust Singapore

Author Contributions

SO, Methodology, Writing- Original draft, Software, Data Curation, validation, and Experiment. SS Conceptualization, Writing- Reviewing and Editing, Supervision. VM Investigation, Writing- Reviewing.

Reference

[1] Varun Chandola, Arindam Banerjee, and Vipin Kumar, "Anomaly Detection: A Survey," *ACM Computing Surveys (CSUR)*, vol. 41, no. 3, pp. 1-58, 2009. [CrossRef] [Google Scholar] [Publisher Link]

[2] Yuan Luo et al., "Deep Learning-Based Anomaly Detection in Cyber-Physical Systems: Progress and Opportunities," *ACM Computing Surveys (CSUR)*," vol. 54, no. 5, pp. 1-36, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[3] Raghavendra Chalapathy, and Sanjay Chawla, "Deep Learning for Anomaly Detection: A Survey," *arXiv*, pp. 1-50, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[4] Charu C. Aggarwal, "An Introduction to Outlier Analysis," *Outlier Analysis, Springer International Publishing*, pp. 1-34, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[5] Tram Truong-Huu et al., "An Empirical Study on Unsupervised Network Anomaly Detection Using Generative Adversarial Networks," *SPAI '20: Proceedings of the 1st ACM Workshop on Security and Privacy on Artificial Intelligence*, pp. 20-29, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[6] Julien Audibert et al., "Usad: Unsupervised Anomaly Detection on Multivariate Time Series," *KDD- 20: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 3395-3404, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[7] Bo Zong et al., "Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection," *International Conference on Learning Representations*, pp. 1-19, 2018. [Google Scholar] [Publisher Link]

[8] Kyle Hundman et al., "Detecting Spacecraft Anomalies Using LSTMs and Nonparametric Dynamic Thresholding," *KDD '18: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 387-395, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[9] Chuxu Zhang et al., "A Deep Neural Network for Unsupervised Anomaly Detection and Diagnosis in Multivariate Time Series Data," *AAAI'19/IAAI'19/EAAI'19: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, vol. 33, pp. 1409-1416, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[10] Eoin Brophy et al., "Generative Adversarial Networks in Time Series: A Survey and Taxonomy," *ACM Computing Surveys*, vol. 55, no. 10, pp. 1-31, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[11] Markus M. Breunig et al., "LOF: Identifying Density-Based Local Outliers," *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, vol. 29, no. 2, pp. 93-104, 2000. [CrossRef] [Google Scholar] [Publisher Link]

[12] Simon D. Duque Anton, Sapna Sinha, and Hans Dieter Schotten, "Anomaly-Based Intrusion Detection in Industrial Data with SVM and Random Forests," *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, Split, Croatia, pp. 1-6, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[13] Asrul H. Yaacob et al., "Arima Based Network Anomaly Detection," *2010 Second International Conference on Communication Software and Networks,* Singapore, pp. 205-209, 2010. [CrossRef] [Google Scholar] [Publisher Link]

[14] Junyoung Chung et al., "Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling," *arXiv*, pp. 1-9, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[15] Muhammad Saad et al., "Tackling Imputation Across Time Series Models Using Deep Learning and Ensemble Learning," *2020 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Toronto, ON, Canada, pp. 3084-3090, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[16] Pankaj Malhotra et al., "LSTM-Based Encoder-Decoder for Multi-Sensor Anomaly Detection," *arXiv*, pp. 1-5, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[17] Zhiwei Ji, Jiaheng Gong, and Jiarui Feng, "A Novel Deep Learning Approach for Anomaly Detection of Time Series Data," *Scientific Programming*, vol. 2021, no. 1, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[18] Daehyung Park, Yuuna Hoshi, and Charles C. Kemp, "A Multimodal Anomaly Detector for Robot-Assisted Feeding Using an LSTM-Based Variational Autoencoder," *IEEE Robotics and Automation Letters*, vol 3, no. 3, pp. 1544-1551, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[19] Ya Su et al., "Robust Anomaly Detection for Multivariate Time Series through Stochastic Recurrent Neural Network," *KDD '19: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2828-2837, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[20] Xuanhao Chen et al., "Daemon: Unsupervised Anomaly Detection and Interpretation for Multivariate Time Series," *2021 IEEE 37th International Conference on Data Engineering (ICDE)*, Chania, Greece, pp. 2225-2230, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[21] Alexander Geiger et al., "TadGAN: Time Series Anomaly Detection Using Generative Adversarial Networks," *2020 IEEE International Conference on Big Data (Big Data)*, Atlanta, GA, USA, pp. 33-43, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[22] Dan Li et al., "MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks," *Artificial Neural Networks and Machine Learning – ICANN 2019: Text and Time Series*, pp. 703-716, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[23] Md Abul Bashar, and Richi Nayak, "TAnoGAN: Time Series Anomaly Detection with Generative Adversarial Networks," *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, Canberra, ACT, Australia, pp. 1778-1785, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[24] Katrina Chen, Mingbin Feng, and Tony S. Wirjanto, "Multivariate Time Series Anomaly Detection via Dynamic Graph Forecasting," *arXiv*, pp. 1-11, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Ailin Deng, and Bryan Hooi, "Graph Neural Network-Based Anomaly Detection in Multivariate Time Series," *Proceedings of the AAAI Technical Track on Data Mining and Knowledge Management*, vol. 35, no. 5, pp. 4027-4035, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[26] Di Ge et al., "An enhanced Spatio-Temporal Constraints Network for Anomaly Detection in Multivariate Time Series," *Knowledge-Based Systems*, vol. 283, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[27] Hang Zhao et al., "Multivariate Time-Series Anomaly Detection via Graph Attention Network," *2020 IEEE International Conference on Data Mining (ICDM)*, Sorrento, Italy, pp. 841-850, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[28] Liwen Zhou, Qingkui Zeng, and Bo Li, "Hybrid Anomaly Detection via Multihead Dynamic Graph Attention Networks for Multivariate Time Series," *IEEE Access*, vol. 10, pp. 40967-40978, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[29] Peihai Zhao, Xiaoyan Chang, and Mimi Wang, "A Novel Multivariate Time-Series Anomaly Detection Approach Using an Unsupervised Deep Neural Network," *IEEE Access*, vol. 9, pp. 109025-109041, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[30] Markus Thill et al., "Temporal Convolutional Autoencoder for Unsupervised Anomaly Detection in Time Series," *Applied Soft Computing*, vol. 112, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[31] Yangdong He, and Jiabao Zhao, "Temporal Convolutional Networks for Anomaly Detection in Time Series," *Journal of Physics: Conference Series*, vol. 1213, no. 4, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[32] Shreshth Tuli, Giuliano Casale, and Nicholas R. Jennings, "TranAD: Deep Transformer Networks for Anomaly Detection in Multivariate Time Series Data," *arXiv*, pp. 1-15, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[33] Jina Kim, Hyeongwon Kang, and Pilsung Kang, "Time-Series Anomaly Detection with Stacked Transformer Representations and 1D Convolutional Network," *Engineering Applications of Artificial Intelligence*, vol. 120, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[34] Ling-rui Yu, Qiu-hong Lu, and Yang Xue, "DTAAD: Dual TCN-Attention Networks for Anomaly Detection in Multivariate Time Series Data," *Knowledge-Based Systems*, vol. 295, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[35] Fanyu Zeng et al., "Multivariate Time Series Anomaly Detection with Adversarial Transformer Architecture in the Internet of Things," *Future Generation Computer Systems*, vol. 144, pp. 244-55, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[36] Alban Siffer et al., "Anomaly Detection in Streams with Extreme Value Theory," *KDD '17: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 1067-1075, 2047. [CrossRef] [Google Scholar] [Publisher Link]

[37] Guangrun Wu, and Wenliang Qiu, "Threshold Selection for POT Framework in the Extreme Vehicle Loads Analysis Based on Multiple Criteria," *Shock and Vibration*, vol. 2018, no. 1, pp. 1-9, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[38] Qingsong Wen et al., "Transformers in Time Series: A Survey," *arXiv*, pp. 1-9, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[39] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun, "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling," *arXiv*, pp. 1-14, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[40] Zhiguo Ding, and Minrui Fei, "An Anomaly Detection Approach Based on Isolation Forest Algorithm for Streaming Data Using Sliding Window," *IFAC Proceedings Volumes*, vol. 46, no. 20, pp. 12-17, 2013. [CrossRef] [Google Scholar] [Publisher Link]

[41] Jonathan Goh et al., "A Dataset to Support Research in the Design of Secure Water Treatment Systems," *Critical Information Infrastructures Security*, *Conference Paper*, pp. 88-99, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[42] Aditya P. Mathur, and Nils Ole Tippenhauer, "SWaT: A Water Treatment Testbed for Research and Training on ICS Security," *2016 International Workshop on Cyber-Physical Systems for Smart Water Networks (CySWater)*, Vienna, Austria, pp. 31-36, 2016. [CrossRef] [Google Scholar] [Publisher Link]