sensors

MDPI

Article

TCF-Trans: Temporal Context Fusion Transformer for Anomaly Detection in Time Series

Xinggan Peng [1], Hanhui Li [2], Yuxuan Lin [1], Yongming Chen [1], Peng Fan [3] and Zhiping Lin [1,*]

[1] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore; xinggan001@e.ntu.edu.sg (X.P.); liny0108@e.ntu.edu.sg (Y.L.); yongming001@e.ntu.edu.sg (Y.C.)
[2] School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518107, China; lihh77@mail.sysu.edu.cn
[3] Chongqing Yuxin Road & Bridge Development Co., Ltd., Chongqing 400060, China; fanpeng@cftecgroup.com
* Correspondence: ezplin@ntu.edu.sg

Abstract: Anomaly detection tasks involving time-series signal processing have been important research topics for decades. In many real-world anomaly detection applications, no specific distributions fit the data, and the characteristics of anomalies are different. Under these circumstances, the detection algorithm requires excellent learning ability of the data features. Transformers, which apply the self-attention mechanism, have shown outstanding performances in modelling long-range dependencies. Although Transformer based models have good prediction performance, they may be influenced by noise and ignore some unusual details, which are significant for anomaly detection. In this paper, a novel temporal context fusion framework: Temporal Context Fusion Transformer (TCF-Trans), is proposed for anomaly detection tasks with applications to time series. The original feature transmitting structure in the decoder of Informer is replaced with the proposed feature fusion decoder to fully utilise the features extracted from shallow and deep decoder layers. This strategy prevents the decoder from missing unusual anomaly details while maintaining robustness from noises inside the data. Besides, we propose the temporal context fusion module to adaptively fuse the generated auxiliary predictions. Extensive experiments on public and collected transportation datasets validate that the proposed framework is effective for anomaly detection in time series. Additionally, the ablation study and a series of parameter sensitivity experiments show that the proposed method maintains high performance under various experimental settings.

Keywords: anomaly detection; deep learning networks; transformer; time series

Citation: Peng, X.; Li, H.; Lin, Y.; Chen, Y.; Fan, P.; Lin, Z. TCF-Trans: Temporal Context Fusion Transformer for Anomaly Detection in Time Series. *Sensors* **2023**, *23*, 8508. https://doi.org/10.3390/s23208508

Academic Editor: Anastasios Doulamis

Received: 20 August 2023
Revised: 26 September 2023
Accepted: 12 October 2023
Published: 17 October 2023

1. Introduction

Anomaly detection aims to find patterns that do not comply with expected behaviour [1]. In many real-world applications, anomaly detection tasks are important research topics [2,3]. Typically, anomalies are categorised as point, contextual, and collective. Since it is common to find that no specific distributions fit the data, and the characteristics of anomalies are different, using traditional anomaly detection methods based on distance estimation or statistical theory may be challenging. Moreover, complex and changing intrinsic data characteristics, low recall rate, and high dimensional data [4] further impede the learning performance of traditional machine learning methods. Under these circumstances, the detection algorithm requires excellent learning ability of the data features. Deep learning methods commonly learn the complex dynamics in the data without relying upon underlying patterns within the data. This advantage makes them popular in dealing with anomaly detection tasks. Transformers, which apply the self-attention mechanism, have shown outstanding performances in modelling long-range dependencies among different deep learning methods.

Common methods for dealing with anomaly detection tasks [5–7] can be generally classified as traditional machine learning methods and deep learning methods.

SVM [8], One-class SVM (OC-SVM) [9], Isolation forest [10] and Local Outlier Factor (LOF) [11] are typical examples of machine learning anomaly detection algorithms. However, if raw samples are complex or dense, the detection performance of these methods will be limited. LODA [12] is a lightweight anomaly detector that ensembles different detectors and is suitable for data streams. LSCP [13] is another ensemble framework compatible with different types of base detectors and further determines the most competent base detector in the local region upon similarity measuring.

Although machine learning methods are suitable for some anomaly detection tasks, deep learning methods more effectively learn expressive representations of complex data in some real-world applications [4,14]. For example, deep support vector data description (DeepSVDD) [15] is applied to complex data for better feature selection. Recurrent neural networks (RNN) that capture time dependence are commonly used to recognise or predict sequences. RNN has been exploited with gating mechanisms to become common methods such as LSTM and Gated recurrent units (GRU). For example, one LSTM-based method is adopted for detecting urban anomalies [16]. DeepAnT [17] is a novel anomaly detection method in time series, which does not require a huge dataset. It primarily applies a CNN-based network to take a window range of time series and try to predict its value for the next stamp. Then, the predicted value is sent to an anomaly detector module to determine its abnormality. DAGMM [18] applies a compression network and an estimation network to achieve unsupervised anomaly detection. The compression network implements a deep autoencoder to generate a low-dimensional representation for each input. Then the estimation network, based on the Gaussian Mixture Model, takes the representation and predicts the corresponding likelihood. Parameters of both the two sub-networks are jointly optimised simultaneously. SO-GAAL [19] applies the generative adversarial learning framework, which consists of a generator and a discriminator used to detect anomalies. Alternatively, GDN [20] combines graph structure learning and attention weights to achieve good anomaly detection results in some fields. LUNAR [21] is another graph neural network-based anomaly detection method. It extracts information from the nearest neighbours of each node and further detects anomalies, and it can learn and adapt to different sets of data. Transformer [22]-based algorithms are also widely applied in anomaly detection tasks. For example, UTRAD [23] obtains stable training and accurate anomaly detection/localisation results based on a transformer-based autoencoder. Additionally, MT-RVAE [24] utilises the variational Transformer model with improved positional encoding and feature extraction to achieve satisfying anomaly detection performances.

In this paper, one Transformer-based network, Informer [25], is chosen as the baseline for dealing with anomaly detection tasks with data collected from real-world applications. The original Informer is an efficient model of Transformer that adopts *ProbSparse* self-attention mechanism to significantly reduce the time complexity and memory usage while outperforming existing methods, mainly in time-series forecasting tasks. Additionally, it can handle tasks in an unsupervised way to avoid the cumbersome labelling cost. However, directly applying the original Informer to time-series anomaly detection tasks may not be appropriate. Since the origin Informer is used in time-series forecasting tasks, it aims to find the overall trend of the target sequence and can ignore some unusual details. Moreover, Transformers may focus on dominant relationships among sequences while paying less attention to intrinsic details when dealing with short-term data. As a result, as shown in Figure 1, it has a straight-throughout feature-transmitting structure for layers in the decoder, and the output of the informer decoder is merely based on the last layer, which contains the least noise but may miss some details. However, anomalies may be rare in anomaly detection tasks, and some minor details in data may reflect the anomaly and cannot be ignored. Different features should be utilised to improve the overall detection performances [26].

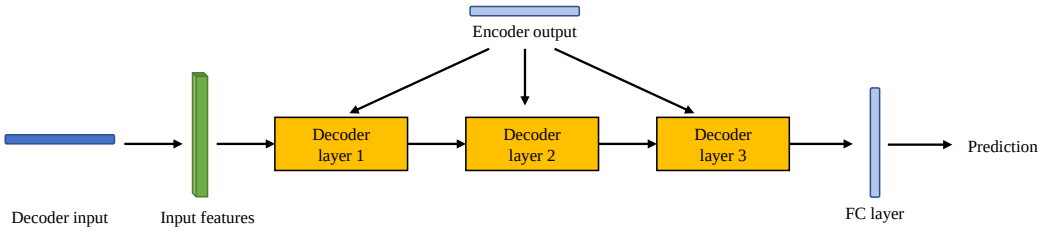


Figure 1. Demonstration example of the 3-layer decoder block diagram for the original Informer network.

To overcome the above-mentioned limitations, we propose to better utilise features from shallow and deep decoder layers with a new multi-layer feature fusion decoder. The original feature-transmitting structure in the decoder of Informer is replaced with the proposed feature fusion decoder to fully utilise the features extracted from shallow and deep decoder layers. This strategy prevents the decoder from missing unusual anomaly details while maintaining robustness from noises inside the data. Next, the auxiliary predictions generated by the decoder will be further adaptively fused based on similarities/distances and sequence information in the temporal context fusion module. This strategy exploits temporal context information of the data by a learnable weight to make the output more robust. We evaluate the proposed method using both the public and our collected transportation datasets for anomaly detection tasks and compare the results with recently proposed machine learning and deep learning methods.

The main contributions of our work can be summarised as follows:

- We introduce a novel framework: Temporal Context Fusion Transformer (TCF-Trans) for unsupervised anomaly detection in time series based on temporal context fusion.
- We replace the straight throughout feature-transmitting structure in the decoder layers of Informer with the proposed feature fusion decoder, which fully utilises the features extracted from shallow and deep decoder layers. This strategy prevents the decoder from missing unusual anomaly details while maintaining robustness from noises inside the data.
- We propose the temporal context fusion module to fuse the auxiliary predictions generated by the decoder adaptively. This strategy alleviates noises or distortions caused by the single auxiliary prediction and fully uses temporal context information of the data.
- Extensive experiments on the public and collected transportation datasets validate that the proposed framework is effective for anomaly detection tasks, such as transportation tasks in time series. In addition, a series of sensitivity experiments and the ablation study show that the proposed method maintains high performance under various experimental settings.

The remaining parts of this paper are organised as follows. Section 2 reviews related works and the background of the Transformer. Section 3 describes the details of the proposed method. Section 4 describes the experiments for validating the proposed method. The conclusion of this paper is presented in Section 5.

2. Related Work

In this section, the background and a review of related works are presented. We first formulate the anomaly detection task in Section 2.1. The background of the transformers and Informer are described in Section 2.2 and Section 2.3, respectively.

2.1. Problem Statement

Given a time-series $\mathbf{x} \in \mathbb{R}^{C \times L}$, where C is the number of channels, and L is the length of the sequence, we can obtain an observation sequence $\mathcal{X}^t$ based on $\mathbf{x}$. The observation at time t with length L_x can be represented as follows:

$$\mathcal{X}^t = \{x_1, \ldots, x_{L_x} \quad | \quad x_i \in \mathbb{R}^{d_x}\}, \tag{1}$$

where x_i is the ith observed values and d_x is the dimension of the observation.

Then, we follow the forecasting-based strategy to generate the corresponding prediction sequence $\mathcal{Y}^t$ with length L_y based on $\mathcal{X}^t$, as follows:

$$\hat{\mathcal{Y}}^t = \{\hat{y}_1, \ldots, \hat{y}_{L_y} \quad | \quad \hat{y}_i \in \mathbb{R}^{d_y}\}, \tag{2}$$

where $\hat{y}_i$ is the ith predicted value and d_y is the dimension of the prediction.

Time series anomaly detection methods aim to determine anomalies in $\mathbf{x}$. Our anomaly detection approach is to generate an anomaly score based on anomaly criteria between $\hat{\boldsymbol{\mathcal{Y}}}$ and its corresponding target $\boldsymbol{\mathcal{Y}}$. Next, the anomaly detection result can be obtained by comparing the anomaly score to a threshold.

2.2. Transformer

Compared with traditional machine learning methods, deep learning methods can capture complex dynamics in the data without making assumptions about the raw data. Among numerous deep learning methods, LSTM [27] and Transformer [22] based algorithms are popular in time series tasks, including anomaly detection. Although LSTM-based methods achieve satisfying performances in some anomaly detection tasks, their highly computationally expensive and inefficient long temporal pattern learning capability still limits their performances. Recently, innovations based on the vanilla Transformer [22] have been widely applied in various fields such as natural language processing (NLP) [28,29], computer vision (CV) [30–33] and time series applications [34–36].

Transformers have shown outstanding performances in modelling long-range dependencies based on the self-attention mechanism and therefore are appealing for time series applications. Moreover, compared with LSTM-based methods, Transformer-based methods are commonly more efficient because of parallel computing [37]. The architecture of a vanilla Transformer framework is shown in Figure 2. The vanilla Transformer consists of an encoder and a decoder, each of which is a stack of N identical layers. Each encoder layer contains two sub-layers, including the multi-head self-attention module and a position-wise fully connected feed-forward network. The residual connection [38] is applied on each sub-layer, and a layer normalisation module [39] is set at the end of each sub-layer. As for the decoder, it has one additional layer between the two sub-layers to perform multi-head attention over the output of the encoder. The residual connection and layer normalisation modules are also implemented in the decoder. Alternatively, a mask is added to the self-attention module to prevent positions from attending to subsequent positions.

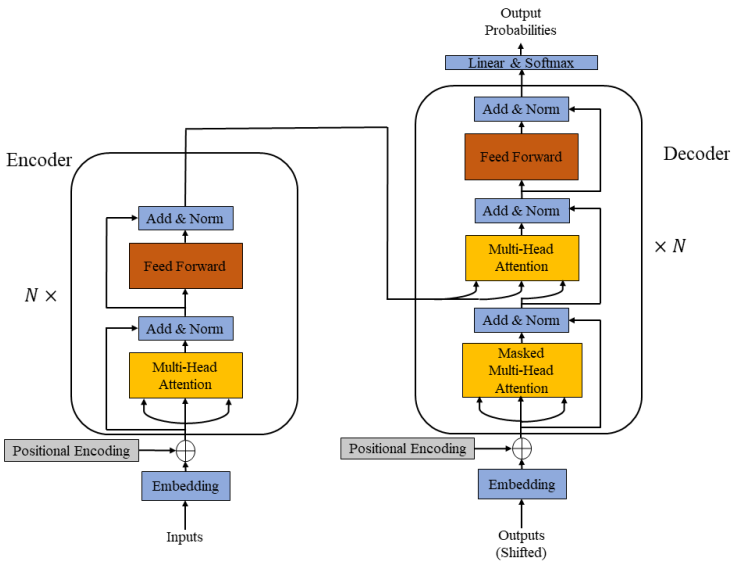


Figure 2. Architecture of vanilla Transformer.

The vanilla Transformer has four key modules: the attention module, feed-forward module, layer normalisation module, and positional encoding module [40].

The attention module takes the scaled dot-product attention with Query-Key-Value (QKV) model, which is shown in Equation (3).

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}(\frac{\mathbf{Q}\mathbf{K}^{\mathbf{T}}}{\sqrt{D_k}})\mathbf{V} \tag{3}$$

where packed matrix representations of queries $\mathbf{Q} \in \mathbb{R}^{L_Q \times D_k}$, keys $\mathbf{K} \in \mathbb{R}^{L_K \times D_k}$, values $\mathbf{V} \in \mathbb{R}^{L_k \times D_V}$ and L_Q, L_K are lengths of queries and keys (or values). D_k, D_V are dimensions of keys (or queries) and values, respectively.

Then, the vanilla Transformer applies the multi-head attention to project the queries, keys and values with H different sets of learned projections, which is shown in Equation (4).

$$\text{MultiHeadAttn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_H)\mathbf{W}^O \tag{4}$$

$$\text{where} \quad \text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_i^Q, \mathbf{K}\mathbf{W}_i^K, \mathbf{V}\mathbf{W}_i^V) \tag{5}$$

where $\mathbf{W}_i^Q$, $\mathbf{W}_i^K$, $\mathbf{W}_i^V$ and $\mathbf{W}^O$ are projection parameter matrices.

The vanilla Transformer performs remarkably in many fields, but quadratic computation complexity remains the bottleneck for some real-world tasks. Recently, Informer [25] selects prototypes from queries using *ProbSparse* mechanism based on Kullback–Leibler divergence between the query's attention probability distribution and the uniform distribution. However, as we mentioned earlier, directly applying the Informer to anomaly detection has limitations. Therefore, a novel temporal context fusion framework named TCF-Trans based on Informer [25] is presented in the next section to better deal with anomaly detection tasks in time series.

2.3. Preliminary on Informer

Although the vanilla Transformer performs remarkably well in many fields, high computational complexity remains challenging for some real-world tasks. Several works have been proposed to improve the efficiency of the vanilla Transformer. Informer [25] is one of the most popular improved versions. It selects prototypes from queries using *ProbSparse* mechanism based on the Kullback–Leibler divergence between the query's attention probability distribution and the uniform distribution. Informer significantly reduces the time complexity and the whole memory usage of the network while maintaining the good learning capability of the Transformer.

The Informer [25] implements a popular encoder-decoder architecture to produce outputs. The encoder encodes the input $\mathcal{X}^t$ into the hidden state representation $\mathcal{H}^t$. Next, the decoder produces output $\hat{\mathcal{Y}}^t$ based on $\mathcal{H}^t$.

The standard self-attention can be replaced by the *ProbSparse* self-attention mechanism to reduce the time complexity from $\mathcal{O}(L^2)$ to $\mathcal{O}(L \ln L)$. This mechanism allows each key to only attend to the Top-k dominant queries, as follows:

$$\mathcal{A}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Softmax}(\frac{\overline{\mathbf{Q}}\mathbf{K}^{\mathbf{T}}}{\sqrt{d}})\mathbf{V}, \tag{6}$$

where $\overline{\mathbf{Q}}$ is the sparse matrix of queries that only contains top-k dominant of queries under the query sparsity measurement [25], d is the dimension of the input, $\mathbf{K}$ is the matrix of keys and $\mathbf{V}$ is the matrix of values.

In the encoder, Self-attention distilling operations can also be implemented between attention blocks to reduce the whole memory usage. The encoded feature will be sent to the decoder layers to produce auxiliary predictions.

As shown in Figure 1, the structure of the Informer decoder contains layer stacks of multi-head attention blocks. The decoder input combines the earlier piece sequence before the output and a placeholder, as follows:

$$\mathbf{X}_{din}^{t} = [\mathbf{X}_{ref}^{t}, \mathbf{X}_{0}^{t}] \in \mathbb{R}^{(L_{ref}+L_{y})\times d_{model}}, \tag{7}$$

where $[\cdot,\cdot]$ donates the concatenation, $\mathbf{X}_{ref}^{t}$ is a L_{ref} long reference sequence, $\mathbf{X}_{0}^{t}$ is a L_{y} long placeholder sequence and d_{model} is the dimension of the model.

When computing *ProbSparse* self-attention in the decoder, a mask can be applied by setting masked dot products to $-\infty$ for inappropriate connections. The final output is produced by a fully connected (FC) layer. The decoder of the Informer utilises multiple layers to extract features and reduce noise, which may be beneficial in forecasting applications, since its goal is to determine the overall trends, and hence minor details may not be necessary. However, since anomalies are rare and some points anomalies differ from the primary trend, this setting becomes inappropriate in anomaly detection tasks. Moreover, different features contain different characteristics, and relying on a single feature may make the method less robust and more likely to be infected by noise in the single feature.

3. TCF-Trans: Temporal Context Fusion Transformer

3.1. Overall Structure

As shown in Figure 3, TCF-Trans consists of three main modules: an auxiliary prediction generator, a temporal context fusion module and an anomaly detection module. The auxiliary prediction generator performs feature learning, fusion and refinement of the processed input data based on an Encoder–decoder architecture. Next, the generated auxiliary predictions are further processed by the temporal context fusion module based on the similarity/distance and sequence information to generate output predictions adaptively. Finally, the output predictions are compared with the target data under anomaly detection criteria to produce anomaly scores. The final detection result will be determined based on the threshold. These modules will be presented in detail in the following sections.

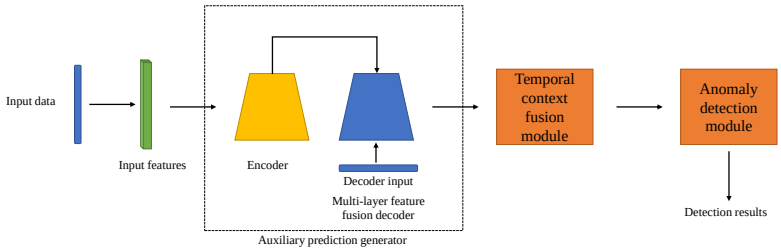


Figure 3. Block diagram of overall structure of TCF-Trans.

3.2. Auxiliary Prediction Generator

The auxiliary prediction generator implements an Encoder–decoder architecture similar to the Informer [25]. The input $\mathcal{X}^{t}$ is encoded into the hidden state representation $\mathcal{H}^{t}$ as the encoder output. Next, the decoder produces output auxiliary predictions $\hat{\boldsymbol{\mathcal{P}}}^{t}$ based on the encoder output $\mathcal{H}^{t}$ and the decoder input $\mathbf{X}_{din}^{t}$.

The process in the encoder is based on the Informer baseline. However, as we mentioned earlier, the process of the Informer baseline in the decoder is ineffective in anomaly detection tasks. To overcome such limitations, we propose a multi-layer feature fusion decoder to better use different features extracted in shallow and deep decoder layers and further refine them to generate auxiliary predictions. To smoothly demonstrate the idea of feature fusion in the decoder, we take a three-layer decoder as an example. As shown in Figure 4, the decoder consists of three layers and inputs are based on encoder output $\mathcal{H}^{t}$ and the decoder input $\mathbf{X}_{din}^{t}$. The outputs for three layers are denoted as $\mathbf{D} = \{\mathbf{d_1}, \mathbf{d_2}, \mathbf{d_3}\}$, where $\mathbf{d_i}$ is the output for the ith layer. Our feature fusion aims to generate the merge and

refined $\mathbf{d_i} \in \mathbb{R}^L$ (denoted as $\mathbf{d_i^*}$) for the ith layer based on $\mathbf{D}$ and use it to produce auxiliary predictions to assist detection.

In the multi-layer decoder, shallow layers contain more information about details, while deep layers contain more depth representations of the data [41,42]. Therefore, we can fuse features from deep layers with those from shallow layers to obtain an optimal representation of the data, as follows:

$$\mathbf{d_i^*} = [\mathbf{d_i}, \ldots, \mathbf{d_1}] \in \mathbb{R}^{L \times d_{feature^*}}, \tag{8}$$

where $[\cdot, \ldots, \cdot]$ donates the concatenation, i is the order of the layer and $d_{feature^*}$ is the dimension of the fused feature.

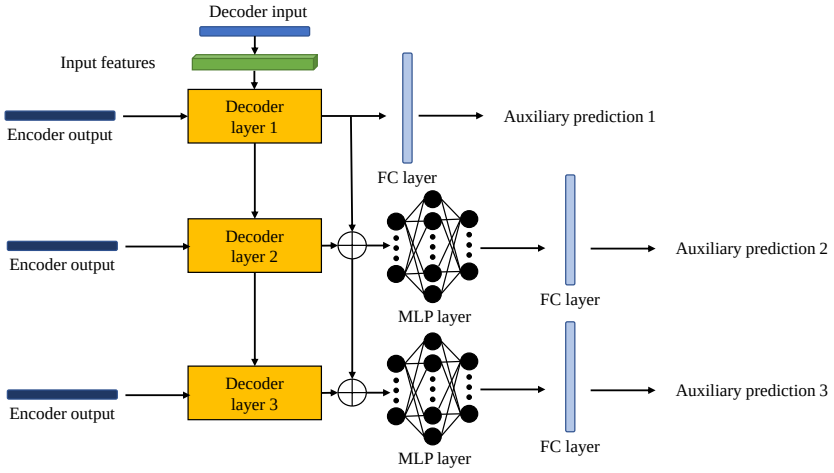


Figure 4. Demonstration example of the three-layer feature fusion decoder of TCF-Trans. $\oplus$ donates feature fusion operations.

During the fusion process, the deeper the layer, the more features it fuses. Under this circumstance, directly applying the FC layer to produce outputs makes it likely to lose information. Therefore, feature-refining tools such as multilayer perceptron (MLP) layers can be implemented to refine the fused features for deeper layers. Note that MLP layers can be represented as hidden layers in the MLP network and layer normalisation can be added to achieve stable transmission.

Lastly, auxiliary predictions produced by the FC layer using refined features can be defined as follows:

$$\hat{\boldsymbol{\mathcal{P}}}^{\boldsymbol{t}} = \{\hat{\mathcal{P}}_1^t, \ldots, \hat{\mathcal{P}}_N^t\}, \tag{9}$$

where N is the number of layers in the decoder and $\hat{\mathcal{P}}_1^t$ is the ith auxiliary prediction.

During the training of this module, the mean squared error (MSE) loss function can be chosen to compute loss among auxiliary predictions and target sequences. Moreover, weights can be assigned for each prediction to emphasise the importance of predictions produced by different layers. The loss function of this module can be expressed as follows:

$$Loss_1 = \sum_{i=1}^{N} w_i MSE(\hat{\mathcal{P}}_i^t, \quad \mathcal{Y}^t), \tag{10}$$

where w_i is the weight of the ith decoder layer.

3.3. Temporal Context Fusion Module and Anomaly Detection

After obtaining several auxiliary predictions produced by the former module, one direct way to combine them is to empirically assign weights to each prediction. However, such a method takes a long time to reach a satisfying solution due to the large number of trials required. Also, applying a scalar weight limited the utilisation of the temporal

context of these predictions. Because different dimensions or points in the sequence may include different intrinsic temporal contexts or the importance of one prediction, applying a scalar weight to a prediction means only different importance exists on different predictions. However, all points in the sequence and dimensions may not share the same importance. In that case, the output prediction based on a fixed scalar weight may only partially take advantage of our feature fusion decoder. Therefore, we aim to produce the final prediction adaptively based on a weight learned from these auxiliary predictions' similarity/differences and fused temporal context.

As shown in Figure 5, auxiliary predictions are sent in the similarity/distance measurement block to determine their similarities/differences. We can calculate the similarities/differences among them as follows:

$$d_{i,j} = k(\hat{\mathcal{P}}_i^t, \hat{\mathcal{P}}_j^t) \qquad i \neq j, \tag{11}$$

where $k(\cdot,\cdot)$ can be a user-defined distance or similarity measurement such as Euclidean distance, etc.

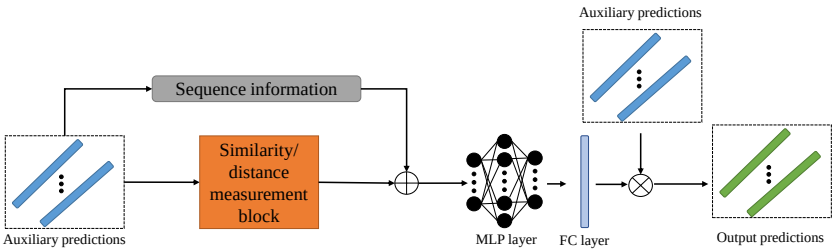


Figure 5. Demonstration example of temporal context fusion module of TCF-Trans. $\oplus$ donates feature fusion operations.

In the meantime, a slice or all of each auxiliary prediction $\triangle\mathcal{P}_i^t \in \mathbb{R}^{L_s \times d_s}$ can be chosen as the sequence information, where L_s and d_s are its length and dimension, respectively. Target sequences could also be added to the sequence information, and we take the auxiliary prediction as an example for convenience of understanding. Then, the temporal context fusion is processed as follows:

$$\hat{\mathcal{D}}^t = [d_{1,2}, \ldots, d_{N,N-1}, \triangle\mathcal{P}_1^t, \ldots, \triangle\mathcal{P}_N^t], \tag{12}$$

where $[\cdot, \ldots, \cdot, \cdot, \ldots, \cdot]$ donates the concatenation and N is the number of layers in the decoder.

Next, the fused temporal context is processed by MLP layers to further feature extraction and refinement. Since we aim to make full use of auxiliary predictions' similarity/differences and temporal information, the output of the MLP will be sent to the FC layer to adaptively generate the weight $\mathcal{W}^{t*}$ based on different weight resolution expectations on various data inputs. (learned weight with a higher dimension presents higher resolution). Specifically, the learned weight $\mathcal{W}^{t*}$ can be chosen from $\mathcal{W}^t = \{\mathcal{W}^t \in \mathbb{R}^N, \mathcal{W}^t \in \mathbb{R}^{d_y \times N}, \mathcal{W}^t \in \mathbb{R}^{L_y \times d_y \times N}\}$.

Last, the output prediction can be computed as follows:

$$\hat{\mathcal{Y}}^t = \sum_{i=1}^{N} \mathcal{W}_i^{t*} \cdot \hat{\mathcal{P}}_i^t. \tag{13}$$

During the training of this module, we can also implement the MSE loss function to compute loss output prediction and target sequences.

After we have obtained the output predictions, the anomaly detection module compares them with the target sequence under specific criteria. Here, we can choose MSE loss to generate the anomaly score as follows:

$$AnomalyScore = MSE(\mathcal{Y}, \hat{\mathcal{Y}}). \tag{14}$$

Once we have the anomaly score, a threshold can be set to determine anomalies. The threshold can be determined empirically or via grid search for better precision, but such methods require extensive trials and are time-consuming. Alternatively, methods based on Streaming Peaks-Over-Threshold (SPOT) [43] can be applied to determine the threshold th.

Therefore, values exceeding th in the $AnomalyScore$ are considered potential anomalies. The main procedures for detecting anomalies via TCF-Trans are summarised in Algorithm 1.

Algorithm 1: Anomaly detection via TCF-Trans

Input: Test Time-series $\mathbf{x}$ and number of layers in feature fusion decoder N.

Obtain T observations as $\mathcal{X}$ from $\mathbf{x}$;
for $t = 1, \ldots, T$ **do**
  `// Processed in the Auxiliary Prediction Generator`
  for $n = 1, \ldots, N$ **do**
    Obtain output of each decoder $\mathbf{D} = \{\mathbf{d_1}, \ldots, \mathbf{d_N}\}$;
  end
  for $n = 1, \ldots, N$ **do**
    $\mathbf{d_i^*} \leftarrow$ using (8);
    if MLP layers *required* **then**
      $\mathcal{P}_n^t \leftarrow$ refine $\mathbf{d_i^*}$ by MLP layers and using FC layer;
    else
      $\mathcal{P}_n^t \leftarrow$ using FC layer;
    end
  end
  `// Processed in Temporal Context Fusion Module`
  for $i, j = 1, \ldots, N \quad i \neq j$ **do**
    $d_{i,j} \leftarrow$ Compute similarities/differences using (11);
  end
  $\hat{\mathcal{D}}^t \leftarrow$ Temporal context fusion using (12);
  $\mathcal{W}^t \leftarrow$ refine $\hat{\mathcal{D}}^t$ by MLP layers and using FC layer;
  $\mathcal{W}^{t*} \leftarrow$ choose resolution expectation on $\mathcal{W}^t$;
  Output prediction $\hat{\mathcal{Y}}^t \leftarrow$ using (13);
  $AnomalyScore \leftarrow$ using (14);
end
Anomaly detection results $\leftarrow$ applying threshold method on $AnomalyScore$;

Output: Anomaly detection results

4. Experiments

In this section, we validate the effectiveness of the proposed anomaly detection framework through several experiments. First, we describe the experimental setup. Then, we evaluate the proposed framework on three public datasets in Section 4.2. Next, we compare the proposed method with several state-of-the-art methods in the real-world transportation traffic dataset in Section 4.3. Moreover, we conduct an ablation study and parameter sensitivity experiments in Sections 4.4 and 4.5, respectively.

4.1. Setup

We follow standard evaluation metrics in anomaly detection tasks, including F_1 score, Precision, and Recall, to evaluate performances as follows

$$\begin{aligned} Precision &= \frac{TP}{FP + TP}, \\ Recall &= \frac{TP}{FN + TP}, \\ F_1 &= 2 \times \frac{Precision \times Recall}{Precision + Recall}, \end{aligned} \tag{15}$$

where TP denotes the number of correct anomalous detections, FP denotes the number of incorrect anomalous detections, and FN denotes the number of incorrect normal detection. A higher F_1 score, precision and recall demonstrate better performances.

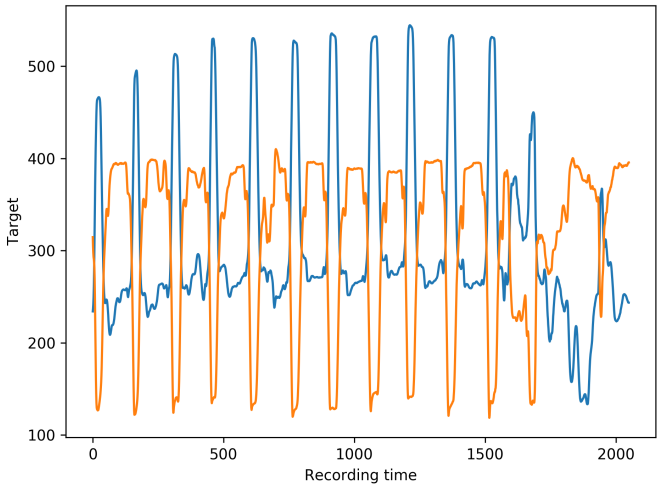
The proposed method is implemented with the Pytorch [44] framework and runs on the NVIDIA RTX 3080 GPU. Some comparison methods are based on publicly available codes provided by PyOD [45]. We implement three layers ($l = 3$) for the feature fusion decoder and the dimension of the model $d_{model} = 512$.

4.2. Evaluation on Public Datasets

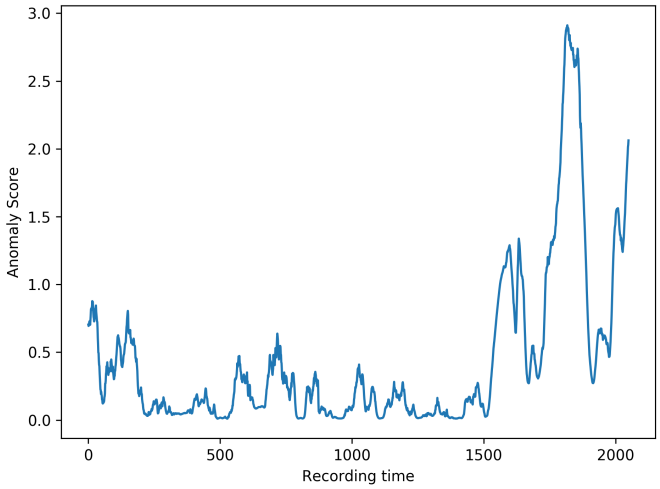
We evaluate the proposed method by applying it to three real-world public anomaly detection datasets. The first public dataset is a gesture dataset [46] collected in a real-world scenario. It records X and Y moving coordinates of one actor's right hand into a time-series sequence. During the actor's actions with the right hand, anomalous actions within a specific time period are recorded. In total, the number of data points in the gesture dataset is around 11,000, with a dimension of two. Around 70% of the samples are used to train the model, while the rest of the data are used for testing. The second and the third public datasets are provided in NAB (The Numenta Anomaly Benchmark) [47] with known anomaly causes collected in the real-world scenario. The second one contains temperature sensor data of an internal component of a large industrial machine (i.e., machine temperature dataset). The third one contains ambient temperature data in an office setting (i.e., ambient temperature dataset). Each dataset has more than 7000 univariate data samples collected in time series. Part of the dataset is used as the training set, while the rest is set for testing.

In the gesture dataset, we implement three state-of-the-art comparison methods, including LUNAR [21], DeepAnt [17], and DeepSVVD [15]. Comparison anomaly detection results on this dataset are shown in Table 1, where bold faced number in each column of the table indicates the best result among all the methods in comparison, which is applied to all other tables in this paper. The results show that the proposed TCF-Trans obtains the best performance in terms of F_1 score, which indicates this method has a good balance in terms of its overall performance without biased detection. Although the recall of the proposed method is not the highest, the methods with higher recall values can suffer from low precision. This phenomenon means they are highly likely to generate false alarms. Therefore, the proposed method obtains competitive performances among the compared state-of-the-art anomaly detection methods. The visualisation examples of detection result slices achieved via the proposed method and LUNAR on the gesture dataset are presented in Figure 6. The X-axis represents recording time. Figure 6a represents the raw data from the test set. Figure 6b and Figure 6c are the corresponding anomaly scores of the proposed method and LUNAR, respectively. Potential anomalies continuously happen after around the 1600th recording time. Compared with LUNAR, the proposed method has more stable anomaly scores among the anomalous regions. On the contrary, anomaly scores obtained from LUNAR suffer from many missing alarms.

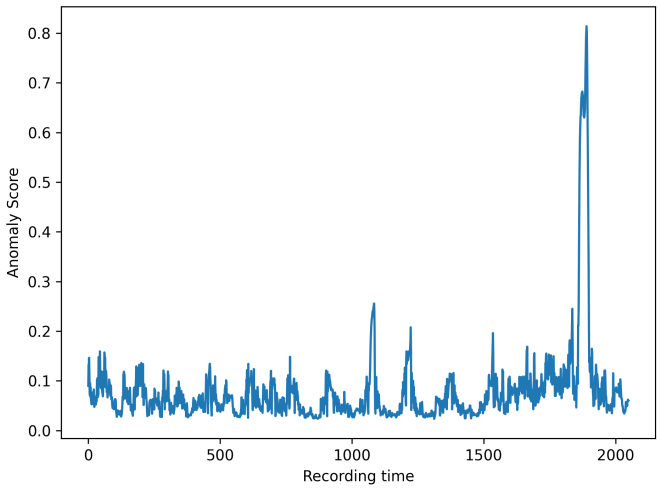
In the machine temperature dataset and the ambient temperature dataset, we implement four state-of-the-art comparison methods, including LSCP [13], LUNAR [21], SO-GAAL [19], and DeepSVVD [15]. Comparison anomaly detection results on the machine temperature dataset are shown in Table 2. Results show that the proposed method outperforms other methods in the F_1 score, with close precision and recall values, indicating that the proposed method can avoid biased detection results. Although some methods have higher recall values than the proposed method, the gap between their recall and precision is noticeable. This unbalanced performance prevents such methods from making accurate predictions among normal samples.



(**a**)



(**b**)



(**c**)

Figure 6. Visualisation examples on the gesture dataset. (**a**) Raw data slice from the test set (**b**) Visualisation example of a detection result slice by the proposed method (**c**) Visualisation example of a detection result slice by LUNAR.

Table 1. Comparison anomaly detection results of the proposed method with other methods on the real-world gesture dataset.

Method	F_1 (%)	Precision (%)	Recall (%)
LUNAR [21]	50.33	38.57	72.40
DeepAnt [17]	40.08	25.07	**99.86**
DeepSVVD [15]	50.00	43.46	58.86
TCF-Trans	**69.69**	**61.67**	80.11

Table 2. Comparison anomaly detection results of the proposed method with other methods on the machine temperature dataset.

Method	F_1 (%)	Precision (%)	Recall (%)
LSCP [13]	60.53	62.30	58.86
LUNAR [21]	45.59	**88.66**	30.69
SO-GAAL [19]	59.76	57.61	**62.08**
DeepSVVD [15]	54.51	81.62	40.92
TCF-Trans	**62.55**	68.76	57.36

Table 3 summarises the comparison anomaly detection results on the ambient temperature dataset. Results show that TCF-Trans obtains satisfying results among the four methods compared here. Meanwhile, we notice that performance vibrations of some methods on these three datasets are apparent, while the proposed method is more stable for different datasets. This advantage indicates that the proposed method is less sensitive to the change in the intrinsic dataset and may be exploited for many detection tasks.

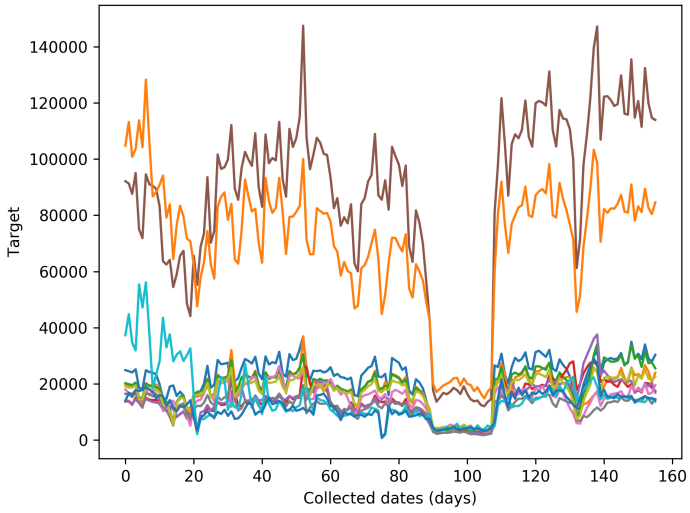
Table 3. Comparison anomaly detection results of the proposed method with other methods on the ambient temperature dataset.

Method	F_1 (%)	Precision (%)	Recall (%)
LSCP [13]	41.24	48.87	35.67
LUNAR [21]	44.33	35.13	60.06
SO-GAAL [19]	40.36	40.50	40.22
DeepSVVD [15]	32.19	20.41	76.03
TCF-Trans	**60.55**	**50.18**	**76.31**

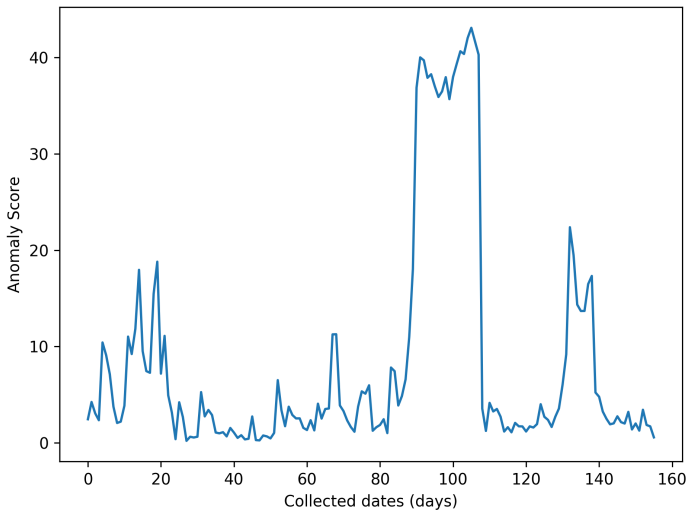
4.3. Evaluation of the Real-World Transportation Dataset

The proposed method is applied to one real-world collected transportation dataset for anomaly detection tasks to show its potential in other real-life applications. The dataset is collected in Chongqing, China, by days (i.e., real-world transportation dataset (days)), and it contains vehicle traffic data from several roads. This dataset with a day collection rate can reflect long-term traffic anomalies, which can be valuable for local enforcement, helping to assess overall traffic planning and management. Moreover, since the relatively large collection rate requires a longer accumulation of data to enlarge its size, the collection difficulty is greater, and the relatively small size of this dataset already contains information for several months. Under such circumstances, the proposed method's learning ability with a small number of samples can also be evaluated. The statistical summary of this dataset is shown in Table 4. In total, the real-world transportation dataset (days) contains 270-day vehicle traffic data from different roads. Part of the dataset, which does not contain an anomaly, is used as the training set, while the rest is chosen as the testing set. We implement five state-of-the-art anomaly detection methods for comparison, including LODA [12], LSCP [13], LUNAR [21], SO-GAAL [19], and DeepSVVD [15]. Table 5 presents the comparison anomaly detection results on this dataset. Results show that TCF-Trans effectively detects anomalies in the real-world transportation dataset. It can be observed that the proposed method balances precision and recall. We owe this good ability to the proposed

feature fusion strategy because we fuse different features with different characteristics. This strategy can alleviate the negative drawbacks of being vulnerable to noises or missing anomaly details. As a result, the proposed method can obtain good overall detection performance. The visualisation example of a detection result slice is shown in Figure 7. In this figure, the X-axis represents collected dates. Figure 7a contains the raw data slice from the test set, and Figure 7b shows the corresponding anomaly score among the test set. SPOT, which is mentioned in Section 3, can be used to determine the threshold without requiring time-consuming grid search trails. It can be observed that there are two types of anomalies in the raw data. For example, a continuous long-term vehicle traffic drop happens around 100th date, which may reflect a continuous anomalous vehicle traffic drop due to large-scale traffic control by local enforcement. The corresponding anomaly score achieved using the proposed method remains continuously high without a clear drop during this region, which reflects that it can effectively deal with this anomalous pattern. Some short-term sharp changes, such as points before the 140th date, may reflect temporary constructions and belong to another type of anomaly. The anomaly scores in this region are also sufficiently stable, showing that the proposed method can effectively detect short-term anomalies.



(**a**)



(**b**)

Figure 7. Visualisation examples on the real-world transportation dataset (days). (**a**) Raw data slice from the test set, where each graph line represents traffic from different raw data sources (**b**) visualisation example of a detection result slice achieved using the proposed method

Table 4. Statistical summary of the real-world transportation dataset (days).

Dataset	Dimension of Data	Training Size	Testing Size
real-world transportation dataset (days)	12	104	166

Table 5. Comparison anomaly detection results of the proposed method with other methods on the real-world transportation dataset (days).

Method	F_1 (%)	Precision (%)	Recall (%)
LODA [12]	83.05	81.67	84.48
LSCP [13]	90.91	96.15	86.21
LUNAR [21]	88.89	96.00	82.76
SO-GAAL [19]	82.64	79.37	86.21
DeepSVVD [15]	87.39	85.25	89.66
TCF-Trans	**93.81**	**96.36**	**91.38**

4.4. Ablation Study

In this section, we conduct the ablation study on the real-world transportation dataset (days) to analyse the effectiveness of each component of the proposed method. We implement four variants of the proposed method, including (i) the Informer baseline, (ii) the TCF-Trans w/o temporal context fusion, in which we replace the temporal context fusion module with one FC layer to generate the output directly, and (iii) the TCF-Trans w/o feature fusion, in which we do not fuse features form different layers. To minimise the impact of different threshold methods, we present results via the grid search based on F_1 score to check performances in theory (with notations †) in this ablation study.

Based on results shown in Table 6, we make the following observations: (1) The proposed TCF-Trans utilises the advantages of each sub-module to achieve the optimal performance among these variants. (2) Since features from different layers have different characteristics, fusing features from different layers helps to improve detection performances. Fusing them can make the proposed method robust to noise and prevent the decoder from missing potential details related to anomalies. (3) Adaptively fusing the auxiliary predictions based on temporal context helps to satisfactorily generate results. On the contrary, directly transforming auxiliary predictions to the final output may be inappropriate.

Table 6. Ablation study results on the real-world transportation dataset (days).

Method	F_1 (%)	Precision (%)	Recall (%)
Informer baseline †	94.02	93.22	94.83
TCF-Trans w/o temporal context fusion †	85.22	85.96	84.48
TCF-Trans w/o feature fusion †	94.74	96.43	93.10
TCF-Trans †	**96.55**	**96.55**	**96.55**

Next, we implement another optimiser SGD on the proposed method to evaluate its impacts. Table 7 shows the results of using different optimisers. The results obtained using SGD are worse than the Adam. The low performance can be led by SGD trapping in the local optimal.

Table 7. Comparison of the anomaly detection performance of the proposed method with different optimisers in relation to the real-world transportation dataset (days).

Optimiser	F_1 (%)	Precision (%)	Recall (%)
SGD †	88.50	90.91	86.21
Adam † *	**96.55**	**96.55**	**96.55**

* Current optimiser for the proposed method.

Moreover, we gradually decrease the number of data used for training from 100% to 80% to show the proposed method's potential to achieve effective detection performances without requiring the accumulation of a large amount of data. The results using different ratios of data used are listed in Table 8. It can be found that F_1 scores do not vary much with ratios from 100% to 80%, which indicates that the proposed method has the potential to obtain satisfying detection performances. Moreover, the results further validate the proposed method's learning ability with few samples.

Table 8. Comparison of the anomaly detection performance of the proposed method with different ratios of training data on the real-world transportation dataset (days).

Ratio of Data (%)	F_1 (%)	Precision (%)	Recall (%)
80 †	92.86	96.30	89.66
90 †	94.74	96.43	93.10
100 † *	**96.55**	**96.55**	**96.55**

* Default ratio of data for the proposed method used in training.

4.5. Parameter Sensitivity Experiments

In this section, the proposed method is implemented under different parameter settings to evaluate its sensitivity to parameter changes on the real-world transportation dataset (days). We also report results achieved via a grid search, in theory (with notations †), to reduce the impact of different threshold methods.

Since the decoder input $\mathbf{X}^t_{din}$ combines the earlier piece sequence before the output and a placeholder, we aim to evaluate the effect of reference sequence $\mathbf{X}^t_{ref}$ with different lengths L_{ref}, as well as corresponding input sequence lengths. Therefore, we choose five reference and input length combinations to evaluate their effect. The results of the proposed method using different combinations of lengths are summarised in Table 9. Our F_1 scores with different lengths are close, which indicates that the proposed method is good at dealing with different input and reference sequences.

Table 9. Comparison of the anomaly detection performance of the proposed method with different lengths of input and reference sequence on real-world transportation dataset (days).

Input and Reference Length	F_1 (%)	Precision (%)	Recall (%)
[3 & 1] †	93.91	94.74	93.10
[5 & 1] †	94.02	93.22	94.83
[5 & 2] †*	**96.55**	**96.55**	**96.55**
[7 & 2] †	93.81	96.36	91.38
[9 & 3] †	91.67	88.71	94.83

* Current lengths of input and reference sequence.

Meanwhile, we aim to determine the impact on a larger number of decoder layers. We increase the number of decoder layers to four (i.e., four-layer TCF-Trans) to validate its performance. We also compare the performances of the Informer baseline with four decoder layers (i.e., four-layer Informer baseline). As shown in Table 10, when comparing performances with the four-layer Informer baseline, the impacts of increasing decoder layers for the proposed method are not serious. This can be explained by our fusion strategy, which fully utilises features from shallow and deep layers to make the method more robust to the noise from a single layer while retaining important details for anomaly detection.

Other loss functions besides the MSE loss may also be chosen as the training loss during the training. We also evaluate the proposed method using two different training loss functions: MAE loss and SmoothL1 loss. The results of using different types of training loss are summarised in Table 11. These results show that F_1 scores with different types of training loss do not vary much, which indicates that loss functions can be adopted in our method.

Table 10. Comparison of the anomaly detection performance of the proposed method with a four-layer decoder on the real-world transportation dataset (days).

Method	F_1 (%)	Precision (%)	Recall (%)
4-layer Informer baseline †	85.22	85.96	84.48
4-layer TCF-Trans †	93.91	94.74	93.10
TCF-Trans †	**96.55**	**96.55**	**96.55**

Based on experiments conducted on the proposed method, the proposed method can handle anomaly detection tasks with different settings, and it shows robustness to these changes. Therefore, the proposed method: TCF-Trans, is an effective solution for anomaly detection in time series applications.

Table 11. Comparison of the anomaly detection performance of the proposed method with different types of training loss on the real-world transportation dataset (days).

Optimiser	F_1 (%)	Precision (%)	Recall (%)
MAE †	94.02	93.22	94.83
SmoothL1 †	95.73	94.92	**96.55**
MSE † *	**96.55**	**96.55**	**96.55**

* Current training loss for the proposed method.

5. Conclusions

This paper has addressed anomaly detection tasks in time series based on a novel framework named temporal context fusion transformer (TCF-Trans). This model utilises the Transform's excellent long-range dependencies modelling capacities and fuses features extracted from shallow and deep decoder layers to prevent the decoder from missing unusual anomaly details while maintaining robustness from noises inside the data, and it improves detection performances. Additionally, the proposed temporal context fusion module fuses the auxiliary predictions generated by the decoder adaptively. It makes the output more robust to noises or distortions caused by the single auxiliary prediction and fully uses temporal context information of the data with a learnable weight. We have performed extensive experiments using the proposed framework on the public and collected transportation datasets. Results have shown that the proposed framework is applicable for some real-world anomaly detection tasks such as transportation in time series. Additionally, the ablation study and several parameter sensitivity experiments have shown that the proposed method can maintain a high performance under various experimental settings.

Author Contributions: X.P. co-designed and implemented the method and prepared the manuscript draft, co-conducted the experiments, and analyzed the experiment results. H.L. helped in designing the algorithm and conducting the experiments. Y.L. helped in processing raw data and conducting the experiments. P.F. helped in collecting and processing raw data. Y.C. and Z.L. provided valuable advice to this work and edited the manuscript drafts. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors would like to express their thanks to Chongqing Yuxin Road & Bridge Development Co., for providing the real-world data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv.* **2009**, *41*, 1–58. [CrossRef]
2. Cherdo, Y.; Miramond, B.; Pegatoquet, A.; Vallauri, A. Unsupervised Anomaly Detection for Cars CAN Sensors Time Series Using Small Recurrent and Convolutional Neural Networks. *Sensors* **2023**, *23*, 5013. [CrossRef] [PubMed]
3. Xu, Z.; Yang, Y.; Gao, X.; Hu, M. DCFF-MTAD: A Multivariate Time-Series Anomaly Detection Model Based on Dual-Channel Feature Fusion. *Sensors* **2023**, *23*, 3910. [CrossRef] [PubMed]
4. Pang, G.; Shen, C.; Cao, L.; Hengel, A.V.D. Deep learning for anomaly detection: A review. *ACM Comput. Surv.* **2021**, *54*, 1–38. [CrossRef]
5. El Sayed, A.; Ruiz, M.; Harb, H.; Velasco, L. Deep Learning-Based Adaptive Compression and Anomaly Detection for Smart B5G Use Cases Operation. *Sensors* **2023**, *23*, 1043. [CrossRef] [PubMed]
6. Kim, B.; Alawami, M.A.; Kim, E.; Oh, S.; Park, J.; Kim, H. A comparative study of time series anomaly detection models for industrial control systems. *Sensors* **2023**, *23*, 1310. [CrossRef]
7. Lan, D.T.; Yoon, S. Trajectory Clustering-Based Anomaly Detection in Indoor Human Movement. *Sensors* **2023**, *23*, 3318. [CrossRef]
8. Fisher, W.D.; Camp, T.K.; Krzhizhanovskaya, V.V. Anomaly detection in earth dam and levee passive seismic data using support vector machines and automatic feature selection. *J. Comput. Sci.* **2017**, *20*, 143–153. [CrossRef]
9. Tian, Y.; Mirzabagheri, M.; Bamakan, S.M.H.; Wang, H.; Qu, Q. Ramp loss one-class support vector machine; A robust and effective approach to anomaly detection problems. *Neurocomputing* **2018**, *310*, 223–235. [CrossRef]
10. Liu, F.T.; Ting, K.M.; Zhou, Z.H. Isolation-based anomaly detection. *ACM Trans. Knowl. Discov. Data TKDD* **2012**, *6*, 1–39. [CrossRef]
11. Mishra, S.; Chawla, M. A comparative study of local outlier factor algorithms for outliers detection in data streams. In *Emerging Technologies in Data Mining and Information Security*; Springer: Singapore, 2019; pp. 347–356.
12. Pevnỳ, T. Loda: Lightweight on-line detector of anomalies. *Mach. Learn.* **2016**, *102*, 275–304. [CrossRef]
13. Zhao, Y.; Nasrullah, Z.; Hryniewicki, M.K.; Li, Z. LSCP: Locally selective combination in parallel outlier ensembles. In Proceedings of the 2019 SIAM International Conference on Data Mining, SIAM, Santa Barbara, CA, USA, 2–4 May 2019; pp. 585–593.
14. Choi, K.; Yi, J.; Park, C.; Yoon, S. Deep learning for anomaly detection in time-series data: Review, analysis, and guidelines. *IEEE Access* **2021**, *9*, 120043–120065. [CrossRef]
15. Ruff, L.; Vandermeulen, R.; Goernitz, N.; Deecke, L.; Siddiqui, S.A.; Binder, A.; Müller, E.; Kloft, M. Deep one-class classification. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 4393–4402.
16. Trinh, H.D.; Giupponi, L.; Dini, P. Urban anomaly detection by processing mobile traffic traces with LSTM neural networks. In Proceedings of the 2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON), Boston, MA, USA, 10–13 June 2019; pp. 1–8.
17. Munir, M.; Siddiqui, S.A.; Dengel, A.; Ahmed, S. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. *IEEE Access* **2018**, *7*, 1991–2005. [CrossRef]
18. Zong, B.; Song, Q.; Min, M.R.; Cheng, W.; Lumezanu, C.; Cho, D.; Chen, H. Deep Autoencoding Gaussian Mixture Model for Unsupervised Anomaly Detection. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
19. Liu, Y.; Li, Z.; Zhou, C.; Jiang, Y.; Sun, J.; Wang, M.; He, X. Generative adversarial active learning for unsupervised outlier detection. *IEEE Trans. Knowl. Data Eng.* **2019**, *32*, 1517–1528. [CrossRef]
20. Deng, A.; Hooi, B. Graph Neural Network-Based Anomaly Detection in Multivariate Time Series. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 4027–4035. [CrossRef]
21. Goodge, A.; Hooi, B.; Ng, S.K.; Ng, W.S. LUNAR: Unifying Local Outlier Detection Methods via Graph Neural Networks. *Proc. AAAI Conf. Artif. Intell.* **2022**, *36*, 6737–6745. [CrossRef]
22. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.U.; Polosukhin, I. Attention is All you Need. In *Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017*; Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
23. Chen, L.; You, Z.; Zhang, N.; Xi, J.; Le, X. UTRAD: Anomaly detection and localization with U-Transformer. *Neural Netw.* **2022**, *147*, 53–62. [CrossRef]
24. Wang, X.; Pi, D.; Zhang, X.; Liu, H.; Guo, C. Variational transformer-based anomaly detection approach for multivariate time series. *Measurement* **2022**, *191*, 110791. [CrossRef]
25. Zhou, H.; Zhang, S.; Peng, J.; Zhang, S.; Li, J.; Xiong, H.; Zhang, W. Informer: Beyond Efficient Transformer for Long Sequence Time-Series Forecasting. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 11106–11115. [CrossRef]
26. Li, H.; Peng, X.; Zhuang, H.; Lin, Z. Multiple Temporal Context Embedding Networks for Unsupervised time Series Anomaly Detection. In Proceedings of the ICASSP 2022—2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Singapore, 23–27 May 2022; pp. 3438–3442.
27. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
28. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **2020**, *21*, 1–67.

29. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding. *arXiv* **2018**, arXiv:1810.04805.
30. Carion, N.; Massa, F.; Synnaeve, G.; Usunier, N.; Kirillov, A.; Zagoruyko, S. End-to-end object detection with transformers. In Proceedings of the European Conference on Computer Vision, Glasgow, UK, 23–28 August 2020; Springer: Cham, Switzerland, 2020; pp. 213–229.
31. Parmar, N.; Vaswani, A.; Uszkoreit, J.; Kaiser, L.; Shazeer, N.; Ku, A.; Tran, D. Image transformer. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm Sweden, 10–15 July 2018; pp. 4055–4064.
32. Chen, H.; Wang, Z.; Tian, H.; Yuan, L.; Wang, X.; Leng, P. A Robust Visual Tracking Method Based on Reconstruction Patch Transformer Tracking. *Sensors* **2022**, *22*, 6558. [CrossRef] [PubMed]
33. Xian, T.; Li, Z.; Zhang, C.; Ma, H. Dual Global Enhanced Transformer for image captioning. *Neural Netw.* **2022**, *148*, 129–141. [CrossRef] [PubMed]
34. Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.X.; Yan, X. Enhancing the Locality and Breaking the Memory Bottleneck of Transformer on Time Series Forecasting. In *Proceedings of the Advances in Neural Information Processing Systems, Vancouver, BC, USA, 10–12 December 2019*; Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc.: Red Hook, NY, USA, 2019; Volume 32.
35. Liu, M.; Ren, S.; Ma, S.; Jiao, J.; Chen, Y.; Wang, Z.; Song, W. Gated Transformer Networks for Multivariate Time Series Classification. *arXiv* **2021**, arXiv:2103.14438.
36. Wang, C.; Xing, S.; Gao, R.; Yan, L.; Xiong, N.; Wang, R. Disentangled Dynamic Deviation Transformer Networks for Multivariate Time Series Anomaly Detection. *Sensors* **2023**, *23*, 1104. [CrossRef]
37. Wen, Q.; Zhou, T.; Zhang, C.; Chen, W.; Ma, Z.; Yan, J.; Sun, L. Transformers in Time Series: A Survey. *arXiv* **2023**, arXiv:2202.07125.
38. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
39. Ba, J.L.; Kiros, J.R.; Hinton, G.E. Layer Normalization. *arXiv* **2016**, arXiv:1607.06450.
40. Lin, T.; Wang, Y.; Liu, X.; Qiu, X. A survey of transformers. *AI Open* **2022**, *3*, 111–132. [CrossRef]
41. Wang, P.; Zheng, W.; Chen, T.; Wang, Z. Anti-Oversmoothing in Deep Vision Transformers via the Fourier Domain Analysis: From Theory to Practice. *arXiv* **2022**, arXiv:2203.05962.
42. Xue, F.; Chen, J.; Sun, A.; Ren, X.; Zheng, Z.; He, X.; Chen, Y.; Jiang, X.; You, Y. A Study on Transformer Configuration and Training Objective. In Proceedings of the 40th International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July 2023.
43. Siffer, A.; Fouque, P.A.; Termier, A.; Largouet, C. Anomaly detection in streams with extreme value theory. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; pp. 1067–1075.
44. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 8026–8037.
45. Zhao, Y.; Nasrullah, Z.; Li, Z. PyOD: A Python Toolbox for Scalable Outlier Detection. *J. Mach. Learn. Res.* **2019**, *20*, 1–7.
46. Keogh, E.; Lin, J.; Fu, A. HOT SAX: Efficiently finding the most unusual time series subsequence. In Proceedings of the Fifth IEEE International Conference on Data Mining (ICDM'05), Houston, TX, USA, 27–30 November 2005; p. 8. [CrossRef]
47. Ahmad, S.; Lavin, A.; Purdy, S.; Agha, Z. Unsupervised real-time anomaly detection for streaming data. *Neurocomputing* **2017**, *262*, 134–147. [CrossRef]