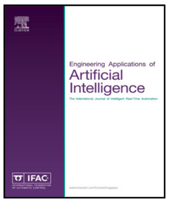
Engineering Applications of Artificial Intelligence 120 (2023) 105964

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Time-series anomaly detection with stacked Transformer representations and 1D convolutional network

Jina Kim[1], Hyeongwon Kang, Pilsung Kang[*]

School of Industrial & Management Engineering, Korea University, 145 Anam-Ro, Seongbuk-Gu, Seoul, 02841, Republic of Korea

ARTICLE INFO

Keywords:
Time series anomaly detection
Transformer
Convolution Neural Network

ABSTRACT

Time-series anomaly detection is a task of detecting data that do not follow normal data distribution among continuously collected data. It is used for system maintenance in various industries; hence, studies on time-series anomaly detection are being carried out actively. Most of the methodologies are based on Long Short-Term Memory (LSTM) and Convolution Neural Network (CNN) to model the temporal structure of time-series data. In this study, we propose an unsupervised prediction-based time-series anomaly detection methodology using Transformer, which shows superior performance to LSTM and CNN in learning dynamic patterns of sequential data through a self-attention mechanism. The prediction model consists of an encoder comprising multiple Transformer encoder layers and a decoder that includes a 1D convolution layer. The output representation of each Transformer layer is accumulated in the encoder to obtain a representation with multi-level, rich information. The decoder fuses this representation through a 1d convolution operation. Consequently, the model can perform predictions considering both the global trend and local variability of the input time-series. The anomaly score is defined as the difference between the predicted and the actual value at the corresponding timestamp, assuming that the trained model produces the predictions that follow the normal data distribution. Finally, the data with an anomaly score above the threshold is detected as an anomaly. Experiments on the benchmark datasets show that the proposed method has performance superior to those of the baselines.

1. Introduction

Time-series anomaly detection is the task of finding data that deviates from the distribution of normal data as determined from the entire time-series (Geiger et al., 2020). Anomaly detection in time-series data collected from a system can be used to monitor the health of the system and predict future problems; thus, it can prevent system failure and reduce system maintenance costs (Li et al., 2021; Martí et al., 2015; Niu et al., 2020). Furthermore, since advances in sensing technology have made it easier to collect time-series data from devices in various fields, such as manufacturing processes, IT, and healthcare, studies on anomaly detection are actively being conducted (Xu et al., 2014; Gao et al., 2020; Marjani et al., 2017).

However, there are some difficulties in anomaly detection in time-series data. First, it is not easy to label anomalies because anomalies seldom occur and need to be defined by comparing them with the overall aspects of the data based on domain knowledge. Therefore, there is very limited labeled data that are available. Second, since time-series anomaly detection needs to be carried out continuously, anomaly detection must be performed promptly for new data by reflecting on the distribution of data that changes over time (Gao et al., 2020). Thus, an appropriate anomaly detection method for time-series is required to learn using only relatively abundant unlabeled data and robust to time-varying patterns. In addition, anomalies should be detected without delay.

Machine learning methods such as local outlier factor (LOF) (Breunig et al., 2000), one-class support vector machine (1-SVM) (Schölkopf et al., 2001), and isolation forest (Liu et al., 2008) have been used to detect anomalies in time-series. However, these methods have a limitation in that they do not consider the temporal structure of the time-series data. To overcome this limitation, deep learning can be utilized. Recently, deep learning has made significant progress in computer vision and natural language processing. As a result, deep learning-based methodologies have also begun to emerge in the field of time-series anomaly detection (Braei and Wagner, 2020). Multiple studies have demonstrated that deep learning-based methods can handle the temporal dependency of time-series better than existing methods based on machine learning (Geiger et al., 2020).

Deep learning-based time-series anomaly detection methods can largely be divided into two categories. First, the prediction-based methods perform anomaly detection by learning the distribution of normal

[*] Corresponding author.
E-mail addresses: jina_kim@korea.ac.kr (J. Kim), hyeongwon_kang@korea.ac.kr (H. Kang), pilsung_kang@korea.ac.kr (P. Kang).
[1] Jina Kim is currently a data scientist at Shinhan Bank, Republic of Korea.

https://doi.org/10.1016/j.engappai.2023.105964
Received 26 April 2022; Received in revised form 13 November 2022; Accepted 4 February 2023
Available online 11 February 2023

data through a model that predicts data at a future point in time using the past data (Malhotra et al., 2015; Munir et al., 2018). This method assumes that the farther the data point moves away from the normal data distribution the prediction model learned, the larger the difference between the actual and predicted values. Hence, this method uses the prediction error for new time-series data as the anomaly score. The reconstruction-based methods detect anomalies by mapping input time-series to a latent space and learning to reconstruct the input time-series from that space (Geiger et al., 2020; Li et al., 2021; Malhotra et al., 2016). This method assumes that precise reconstruction is difficult when input time-series include anomalies because the information on rare anomalies in the training data is lost in the latent space. Hence, the method uses the reconstruction error as the anomaly score. The methods mentioned above mainly use convolution neural networks (CNN) or recurrent neural networks (RNN) to model the temporal structure of time-series data. However, a CNN learns by focusing on local information, and an RNN has a long-term dependency problem. Thus, both structures have the problem of not sufficiently learning global information of the input sequence. Since this issue can be the main cause of performance degradation in an anomaly detection task, which needs to reflect on various patterns, a model that can learn the global information of time-series is needed.

Recently, studies that utilize Transformer (Vaswani et al., 2017) to improve the aforementioned problems of CNN and RNN have attracted attention in the field of time-series data analysis (Wu et al., 2020, 2021; Zhou et al., 2021; Xu et al., 2022). Transformer is a model proposed for natural language processing to effectively learn the global dependency in a sequence through a self-attention mechanism. Therefore, in this study, we propose a stacked Transformer representation and a one-dimensional convolutional network (STOC), a prediction-based unsupervised time-series anomaly detection method utilizing Transformer. Most deep models use only the top-layer representation and ignore information from the intermediate layers (Wang et al., 2018; Ma et al., 2019; Fang et al., 2020). To fully utilize the potential information of lower layers, we stack all output representations from each Transformer encoder layer. The stacked representation contains both (1) the relatively local information from the lower layers and (2) the global information from the upper layers. Subsequently, the multi-level information is effectively combined via a convolution layer. Finally, the last layer predicts the future time-series from the given time-series using the learned representation in earlier layers. Then, anomaly detection is performed based on the prediction error of each timestamp. Experiments performed on 367 time-series of the Yahoo S5 benchmark show that the proposed method yielded higher detection performance than both the existing prediction-based method and reconstruction-based method. In addition, the proposed model was found to be more robust to the data characteristics.

The rest of this paper is structured as follows. In Section 2, we review time-series anomaly detection methods and studies utilizing Transformer for time-series data analysis. Section 3 provides details about the proposed method. In Sections 4 and 5, we present the experimental setting and analyze the result of experiments qualitatively and quantitatively. Section 6 we explore the ablations of the proposed method. Finally, in Section 7, we conclude the current work with some future research directions.

2. Related works

2.1. Prediction-based time-series anomaly detection methods

The prediction-based method predicts future time-series using the fitted model on the training data and detects anomalies based on the difference between the predicted value and the actual value. RNN and CNN are commonly used as prediction models to deal with the temporal structure of time-series. Malhotra et al. (2015) proposed stacked long short-term memory networks (LSTM) (Hochreiter and Schmidhuber, 1997) as the prediction model for anomaly detection (LSTM-AD). The multivariable Gaussian distribution following the prediction error of the training data was estimated, and the likelihood of the prediction error of the new data was utilized as an anomaly score. Later, Munir et al. (2018) proposed a deep learning approach for unsupervised anomaly detection in time-series (DeepAnT) using the convolution operation. This model learns the normal pattern of time-series by applying a deep CNN model, which was constructed by stacking convolution and pooling layers. In contrast to Malhotra et al. (2015), this model used the Euclidean distance between the predicted value and the actual value as the anomaly score and achieved the best performance on most of the ten anomaly detection benchmarks, including the Yahoo S5 benchmark.

2.2. Reconstruction-based time-series anomaly detection methods

The reconstruction-based method is based on the structure that maps the given time-series into a latent vector and then reconstructs it into the original space. Since the model can learn the normal class sufficiently with the given abundant normal data but cannot learn the anomalies due to the lack of abnormal samples in the training dataset, information about anomalies is lost during training. Therefore, abnormal data has a high reconstruction error and then can be detected. In Malhotra et al. (2016), the distribution of normal data was learned using an autoencoder structure consisting of an encoder that compresses the input data into a low-dimensional latent space and a decoder that reconstructs the data from the compressed representation. Li et al. (2019) presented a smoothness-inducing sequential variational autoencoder (SISVAE). This model performed reconstruction of the training data through the sequential variational autoencoder (VAE), whereby the encoder learns the probability distribution of the training data, and the decoder reconstructs the data using the representation sampled from the distribution learned by the encoder. In addition, learning a distribution that changes due to the anomalies included in the training data was regulated by adding smoothness loss to the loss function of the standard sequential VAE. Geiger et al. (2020) proposed time-series anomaly detection using generative adversarial networks (TadGAN), which learn the distribution of normal data effectively through adversarial learning of the generator that generates data following the distribution of the normal data and the discriminator that distinguishes the generated data from real data. All the methods mentioned here detect anomalies based on the reconstruction error of the input data. Moreover, these methods use the encoder, decoder, generator, and discriminator consisting of LSTMs to learn the temporal structure of time-series data.

2.3. Transformer for sequential data analysis

Transformer is a method proposed for machine translation to make it possible to learn global dependencies within a sequence using only the attention mechanism. In addition to machine translation, Transformer has outperformed RNNs and CNNs in handling sequential data in many other natural language processing tasks. Recently, it has been demonstrated that Transformer can also effectively handle sequential data in other fields, such as speech recognition and computer vision (Dosovitskiy et al., 2021; Gulati et al., 2020; Huang et al., 2018). Many studies using Transformer appeared for time-series analysis, which also deals with sequential data. In particular, studies such as Wu et al. (2020) and Zhou et al. (2021) focus on forecasting tasks. These studies predict multi-step time-series by utilizing the high prediction capacity of Transformer, and their experiments show that Transformer outperformed the existing LSTM-based methods in long sequence time-series forecasting (LSTF).

2.4. The advantage of fusing multi-level information in deep learning

Some studies in various fields improve performance by fusing multi-layer information of the deep learning model. Wang et al. (2018),

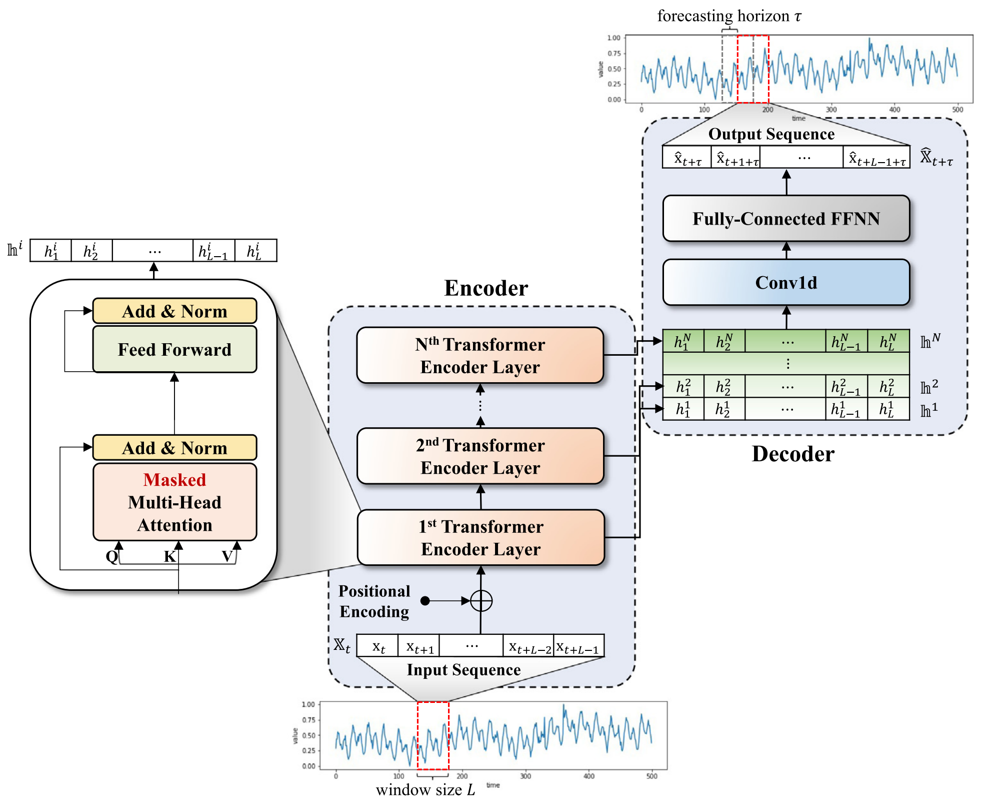

forecasting horizon τ
Output Sequence
Fully-Connected FFNN
Conv1d
Encoder
Decoder
Nth Transformer Encoder Layer
2nd Transformer Encoder Layer
1st Transformer Encoder Layer
Add & Norm
Feed Forward
Masked Multi-Head Attention
Q K V
Positional Encoding
Input Sequence
window size L

Fig. 1. Architecture of the proposed model.

Ma et al. (2019), and Fang et al. (2020) pointed out that focusing only on the top-layer representation leads to loss of meaningful information in lower layers. To prevent this loss and enrich representations, they fused all representations from each layer in a deep model. For example, Wang et al. (2018) proposed a fusion function that densely connects the stacked representations from multi-layers in the encoder and decoder. It was confirmed experimentally that their fusion function improved the regularization performance and achieved a new state-of-the-art in the German–English machine translation task. Fang et al. (2020) extracted the representation by integrating several layers of the feature maps and then delivering it to the classifier, unlike the existing approaches that use only the last convolution layer's feature maps to classify remote sensing scenes. The experimental result proved that the multi-level information integration enhanced the feature representation of input images more than using the information of only the top layer. As mentioned in Section 2.2, reconstruction-based methods can only conduct reconstruction after mapping the new time-series into a latent space. On the other hand, prediction-based methods can predict future time-series in advance; hence, they can immediately calculate the prediction error when new time-series is given and quickly determine whether the new data is abnormal. Therefore, prediction-based methods have an advantage over reconstruction-based methods in terms of the efficiency of anomaly detection. In this study, we take advantage of the benefits of the efficient inference process of the prediction-based method and Transformer, which has outstanding capability to learn the global dependency in a sequence. In addition, we enhance the feature representations of the encoder by stacking multi-level representations of Transformer.

3. Proposed method

3.1. Time-series prediction by combining multi-level representations from transformer layers

In this study, we propose a time-series anomaly detection method based on a prediction model. The prediction model consists of an encoder that learns a rich multi-level representation from multiple Transformer encoder layers and a decoder that effectively combines that representation. The architecture of the prediction model of the proposed method is shown in Fig. 1. The prediction model minimizes the prediction error between the predicted sequence and the actual sequence to learn the distribution of normal data, which predominantly appears in the training data. The given time-series data must be transformed into a suitable form for continuous time-series prediction. Therefore, in this study, multiple time windows are created by splitting the input time-series $\mathbb{X}_{total} = \{\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_T\}$, *where* $\mathbf{x}_l \in \mathbb{R}^{d_x}$, into sequences of window length L. We make overlapped windows using a sliding window mechanism with a window length of L and a step size of 1. The time window of length L that begins at time t is expressed as $\mathbb{X}_t = \{\mathbf{x}_t, \mathbf{x}_{t+1}, \dots, \mathbf{x}_{t+L-1}\}$.

3.1.1. Encoder

When the sequence $\mathbb{X}_t = \{\mathbf{x}_t, \mathbf{x}_{t+1}, \dots, \mathbf{x}_{t+L-1}\}$, *where* $\mathbf{x}_l \in \mathbb{R}^{d_x}$, with window length L is input into the encoder, positional encoding is added in the same manner as Transformer in Vaswani et al. (2017) to provide temporal information on the input sequence. As shown in Eqs. (1) and (2), the input sequence passes through N Transformer encoder layers, whereby a feature representation $\mathbb{h}^i$ is obtained for each layer.

$$\mathbb{h}^1 = TransformerEncoder^1(\mathbb{X}_t) = \{h_1^1, h_2^1, \dots, h_L^1\} \tag{1}$$

$$\mathbb{h}^i = TransformerEncoder^i(\mathbb{h}^{i-1}) = \{h_1^i, h_2^i, \dots, h_L^i\} \quad where\ i \geq 2 \tag{2}$$

where $h_t^i \in \mathbb{R}^d$, and d refers to the dimension of the Transformer encoder. Therefore, $\mathbb{h}^i \in \mathbb{R}^{d \times L}$. Each Transformer layer learns the dependency between data at each timestamp in the input sequence through the following multi-head self-attention operation. Here, unlike the existing Transformer encoder for natural language processing, masked multi-head self-attention is used to reflect the characteristics of the time-series, which has no information on a future timestamp at the

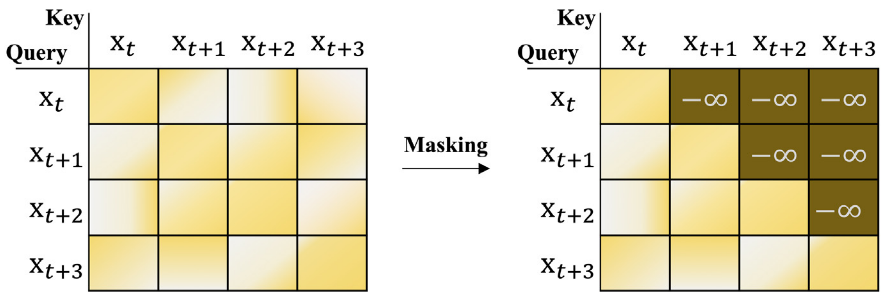


Fig. 2. Masked attention matrix.

current time. As shown in Fig. 2, this procedure forces the Transformer layers to refer only to the past and present timestamps to prevent future information leakage by masking the upper triangular element of the attention matrix with $-\infty$.

To utilize all of the feature representations of different levels learned by each Transformer layer, these feature representations are stacked in the order of the layers to define the final output representation $\mathbb{h}^{encoder\_output}$ of the encoder.

$$\mathbb{h}^{encoder\_output} = stack[\mathbb{h}^1, \mathbb{h}^2, \dots, \mathbb{h}^N] \in \mathbb{R}^{(d\dot{N})\times L} \tag{3}$$

Thus, unlike the existing Transformer that only uses the final output representation of the encoder, this study considers both the local variability and overall trend within a sequence by combining the information from the lower layer – which has learned relatively local information – with the information from the upper layer, which has learned the global dependency.

3.1.2. Decoder

The decoder is designed to receive the encoder output representation $\mathbb{h}^{encoder\_output}$ and combine these multi-level information to predict the sequence – with the same length as the input sequence – τ time steps into the future $\hat{\mathbb{X}}_{t+\tau} = \{\hat{\mathbf{x}}_{t+\tau}, \hat{\mathbf{x}}_{t+1+\tau}, \dots, \hat{\mathbf{x}}_{t+L-1+\tau}\}$, *where* $\hat{\mathbf{x}}_l \in \mathbf{R}^{d_x}$, *and* $\tau \le L$. A 1-dimensional convolution neural network (1D CNN) is used to combine the hierarchical information received from the encoder. The same number of feature maps as the dimension d of the Transformer layer is obtained through the 1D CNN layer, and the representation $\mathbb{h}^{conved}$, which follows the dimension $d\times L$ of the encoder output representation, is learned. This process is described by Eq. (4).

$$\mathbb{h}^{conved} = Conv(\mathbb{h}^{encoder\_output}) \in \mathbb{R}^{d\times L} \tag{4}$$

The 1D CNN is capable of modeling the trend and cycle of a time-series with fewer parameters than RNN (Munir et al., 2018), and each filter learns the same weight that is not affected by time. Hence, 1D CNN is appropriate for modeling the time-invariant characteristics of a time-series. Therefore, the prediction model is designed to effectively learn the common cycle from highly volatile data such that an instance that deviates from this cycle can be easily determined as an anomaly. Finally, a sequence of d_x-dimensional vectors, which is equal to the dimension of the input sequence, is derived from $\mathbb{h}^{conved}$ using a fully-connected layer.

$$\hat{\mathbb{X}}_{t+\tau} = Linear(\mathbb{h}^{conved}) \in \mathbb{R}^{d_x\times L} \tag{5}$$

3.2. Loss function

The prediction model designed as described in Section 3.1 is trained by using the mean squared error (MSE) of the actual value $\mathbb{X}_{t+\tau} = \{\mathbf{x}_{t+\tau}, \mathbf{x}_{t+1+\tau}, \dots, \mathbf{x}_{t+L-1+\tau}\}$ and combined with the predicted value $\hat{\mathbb{X}}_{t+\tau} = \{\hat{\mathbf{x}}_{t+\tau}, \hat{\mathbf{x}}_{t+1+\tau}, \dots, \hat{\mathbf{x}}_{t+L-1+\tau}\}$ for N total number of sequences in the training data.

$$Loss(\mathbb{X}_{t+\tau}, \hat{\mathbb{X}}_{t+\tau}) = \frac{1}{L}\frac{1}{d_x}\sum_{l=1}^{L}\sum_{m=1}^{d_x}(x_{m,t+l-1+\tau} - \hat{x}_{m,t+l-1+\tau})^2 \tag{6}$$

$$Total\ Loss = \frac{1}{N}\sum_{n=1}^{N} Loss(\mathbb{X}^n_{t+\tau}, \hat{\mathbb{X}}^n_{t+\tau}) \tag{7}$$

As in Eq. (7), the prediction model, which has been trained to minimize the difference between the predicted value and the actual value, can learn the distribution of normal data from the training data. Consequently, the model produces predictions that follow the distribution of normal data of given time-series.

3.3. Anomaly detection

In this study, the prediction error for each timestamp is defined as the anomaly score such that an instance where the anomaly score is larger than the threshold is defined as an anomaly. Here, Euclidean distance given in Eq. (8) is used as a metric to measure the prediction error.

$$Anomaly\ Score\ of\ \mathbf{x}_t = s_t = \sum_{m=1}^{d_x}\sqrt{(x_{m,t} - \hat{x}_{m,t})^2} \tag{8}$$

As shown in Fig. 3, time-series data of the most recent L timestamps $\mathbb{X}_{t-L+1} = \{\mathbf{x}_{t-L+1}, \mathbf{x}_{t-L+2}, \dots, \mathbf{x}_t\}$ are used as the input sequence of the trained prediction model in an online anomaly detection scenario. Then the model predicts the sequence at τ time steps into the future $\hat{\mathbb{X}}_{t-L+1+\tau} = \{\mathbf{x}_{t-L+1+\tau}, \mathbf{x}_{t-L+2+\tau}, \dots, \mathbf{x}_{t+\tau}\}$. The anomaly score for new data during τ times is calculated in real time based on the last τ predicted values of the predicted sequence. Therefore, we can use time windows of length L with a step size of 1 to τ in the inference phase. The time windows for inference are overlapped except when τ is equal to L. If the anomaly score exceeds a certain threshold, the data at the corresponding timestamps are considered anomalies. This way, it is possible to detect anomalies in real-time data by performing predictions based on the most recent sequence of length L at every τ step.

4. Experimental setting

4.1. Dataset

In this study, experiments are conducted on the Yahoo S5 and NeurIPS-TS benchmark to verify the effectiveness of the proposed method. First, the Yahoo S5 consists of 367 time-series data points divided into four sub-benchmarks, from A1 to A4. The time-series data belonging to the A1 benchmark is real-world data generated by the Yahoo membership login system (Laptev et al., 2015). The data were collected every hour, and human-annotated anomalies constituted about 1.9% of the entire data. The time-series data belonging to the remaining three benchmarks are synthetic data that have been designed to display various trends and seasonality. In particular, change points exist in the time-series belonging to the A3 and A4 sub-benchmarks. A change-point refers to a particular timestamp where the distribution of the data changes. Anomalies are defined by randomly inserting data with a pattern different from normal data into all benchmarks at particular timestamps. All four sub-benchmarks had timestamp,

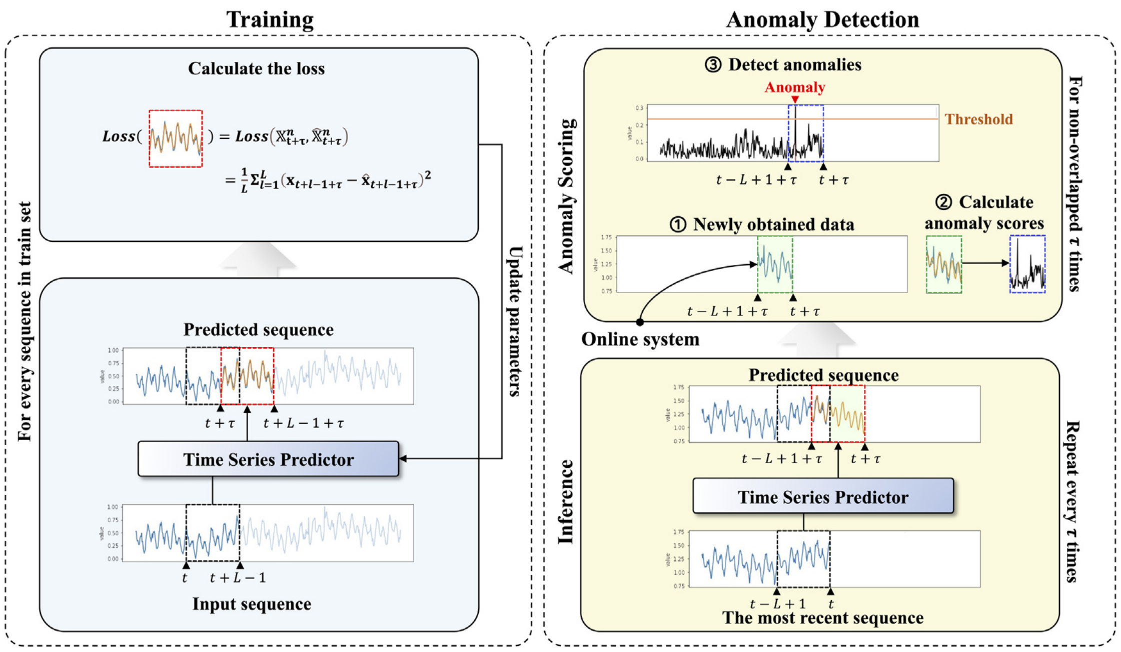


Fig. 3. Training and anomaly detection.

Table 1
Yahoo S5 benchmark summary.

	A1	A2	A3	A4
Real/Synthetic	Real	Synthetic	Synthetic	Synthetic
# of time-series	67	100	100	100
# of data points	94,866	142,100	168,000	168,000
Average length	1,462	1,421	1,680	1,680
# of anomalies	178	200	939	835

value, and the presence of anomaly as features. Sub-benchmarks A3 and A4 had additional features: change-point, trend, noise, and seasonality. However, we did not use features other than the value for training and inference because the information about these features cannot be known in advance. The detailed information on each class is shown in Table 1. Second, we used NeurIPS 2021 Time Series Benchmark (NeurIPS-TS) (Lai et al., 2021). NeurIPS-TS includes point-global, point-contextual outlier, collective-global, collective-seasonal, and collective-trend anomalies. In this paper, we generated 20 time series with anomalies that follows the mechanism of NeurIPS-TS. We set the length to 1500 and the anomaly ratio α from 0.05 to 0.2 by 0.05.

Since this study aimed to detect anomalies based on unsupervised learning, the presence of anomaly, which is the label of the data, was not utilized during preprocessing. Therefore, we did not remove the anomalies in the training data. Furthermore, preprocessing, such as detrending and deseasonalizing were not performed, excluding min–max normalization which was performed on the entire data based on the training data, as shown in Eq. (9)

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}} \tag{9}$$

Of the total data, 30% and 10% were set aside as the training and validation sets, respectively. The remaining 60% were used as test data. This experimental setting is the most conservative setting among the ratios of training, validation, and test of other baselines.

4.2. Baselines and evaluation metrics

In this paper, prediction-based models LSTM-AD (Malhotra et al., 2015) and DeepAnT (Munir et al., 2018) and reconstruction-based models TadGAN (Geiger et al., 2020) and SISVAE (Li et al., 2019) were used for comparisons with the proposed method. These models generally exhibited good performance on the Yahoo S5 benchmark, and they achieved the highest performance based on the F1-score or the area under the ROC curve (AUROC) on at least one sub-benchmark except for LSTM-AD. We additionally compare the proposed model with Transformer-based models, Informer (Zhou et al., 2021), and Anomaly Transformer (Xu et al., 2022). The former is for time series forecasting, and the latter is a reconstruction-based anomaly detection method. Informer follows the detection mechanism of the proposed method. We evaluate the performance of baselines and the proposed method using the AUROC and best F1-score. These metrics are commonly used in time-series anomaly detection.

$$F1 - score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{10}$$

AUROC is defined as the area of the region under the receiver operating characteristic (ROC) curve, which expresses the relationship between the true positive rate and the false positive rate according to the threshold. AUROC can be interpreted as the probability that the method gives a higher anomaly score to a random abnormal data point than a random normal data point in a given time-series. Hence, AUROC is robust against the threshold and appropriate for evaluating anomaly detection methods.

4.3. Implementation details

We constructed an encoder consisting of a positional encoding layer and three Transformer layers and a decoder composed of a 1D convolution layer and two fully-connected layers. Here, the hyperparameters of the networks forming the encoder and decoder were shown in Table 3. In addition, we set the length of input sequences and conducted grid search on the forecasting horizon $\tau = [1, 4, 8, 12, 16, 20, 24]$. For training, the initial learning rate was set to 10^{-5}, and the AdamW optimizer (Loshchilov and Hutter, 2019) was used with a cosine annealing scheduler. All experiments were implemented with PyTorch and were performed in an environment consisting of Intel Core i7-11700X, 128-GB RAM, NVidia GeForce GTX 2080 Ti GPU. The training time of STOC is shown in Table 2. We averaged the training time for 67 (A1) or 100 (A2-4) time series belonging to each sub-benchmark.

Table 2
Training time per each epoch on Yahoo S5 benchmark.

Method	Yahoo S5 (Training time s/epoch)			
	A1	A2	A3	A4
STOC (ours)	2.0022	2.1167	2.2875	2.2833

Table 3
Hyperparameters of the proposed method.

Hyperpatameter			Value
Window length L			48
Encoder	Transformer encoder layer	Feature size d	256
		# of heads	8
		# of layers N	3
Decoder	1dConv	Kernel size	3
		Padding	1
		Stride	1
	# of linear layers		2

4.4. Data augmentation

When the training data are limited, data augmentation is an effective method that can enhance the performance of the model by preventing overfitting to training data (Gao et al., 2020; Wen et al., 2021). In particular, in the case of the Yahoo S5 benchmark, the number of sequences generated from the training data is 440 to 500. Hence, they are not sufficient to learn the time-varying distribution of given time-series.

Therefore, we augmented the training data by shifting each sequence to develop a robust model against the distribution shifts. When the input sequence is $\mathbb{X}_t = \{\mathbf{x}_t, \mathbf{x}_{t+1}, \dots, \mathbf{x}_{t+L-1}\}$, the augmented sequence is $\mathbb{X}_{shifted} = \{\mathbf{x}_t + \delta, \mathbf{x}_{t+1} + \delta, \dots, \mathbf{x}_{t+L-1} + \delta\}$. we set the shift value δ from -3 to 3 at intervals of 0.5.

5. Experimental results

5.1. Quantitative results

In Section 5.1, we compared the performances of the prediction-based and reconstruction-based methods mentioned in Section 4.2 with the performance of STOC using quantitative measures to verify the effectiveness of STOC. The performances of the proposed method and the baselines are described in Table 4. We named the proposed method with a τ forecasting horizon as STOC-τ. We measured the performance of each method on the test set. The STOC outperformed baselines on all sub-benchmarks of Yahoo S5 based on AUROC. Furthermore, the STOC showed better performance than the baseline on the A1 and A2 benchmarks based on the best F1-score. Based on these results, it can be concluded that the method proposed in this study had robust performance in terms of data characteristics and thresholds. Thus, the proposed method can effectively detect anomalies in a time-series with various characteristics. Especially, The STOC-24 had the best performance on the A1 benchmark, which is the real-world data. It shows an average AUROC improvement of 8.8% compared to the three LSTM or CNN baselines. Therefore, we can expect that the proposed method will work well in a real-world application.

Fig. 4 compares the AUROC of the LSTM and CNN-based baselines and that of the STOC. For STOC, we show the performance of the best forecasting horizon in each sub-benchmark. The STOC, represented by the red color, had the best performance. In particular, STOC showed more improved performance than LSTM-AD and DeepAnT, which are both prediction-based methods. This result can be interpreted as STOC using Transformer to model the global and local information within a sequence more effectively than the existing prediction-based methods; hence, STOC effectively produced the predicted values following normal data distribution.

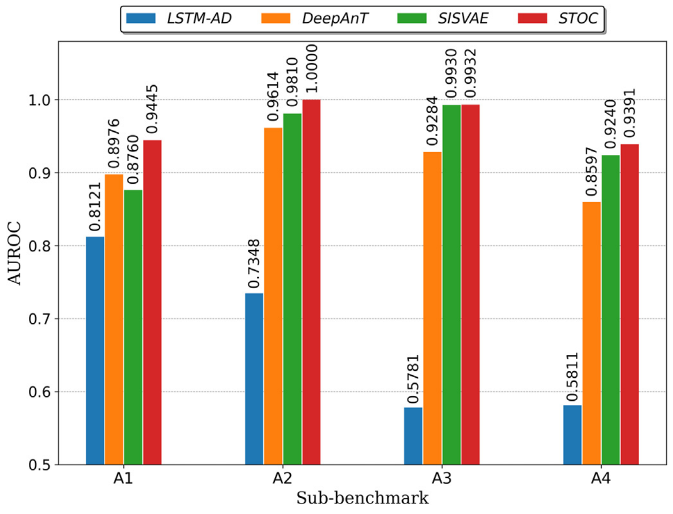


Fig. 4. AUROC comparison baselines and the proposed method with the best forecasting horizon on Yahoo S5 benchmark.

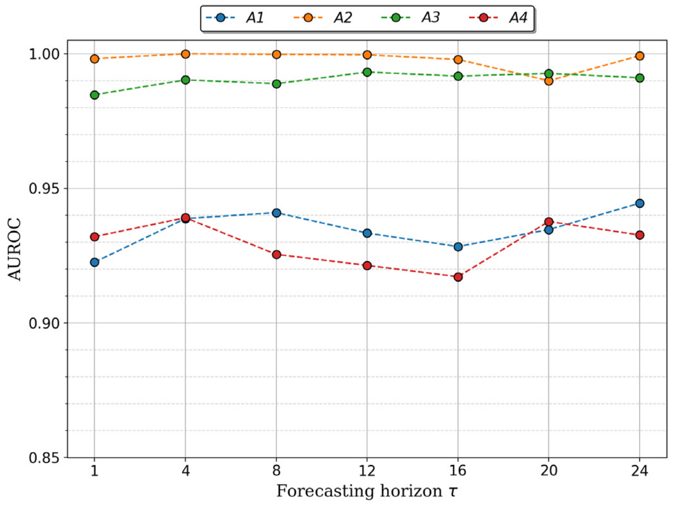


Fig. 5. Performance of the proposed method with various τs on Yahoo S5 Benchmark.

Table 5 shows the average and standard deviation of the AUROC according to the forecasting horizon τ of STOC. Table 5 also shows the performance of the method with the best performance for each sub-benchmark among the LSTM and CNN-based baselines. Here, the AUROC values are indicated with the first letter of the corresponding method. When the average performance of the proposed method was compared with that of the best baseline, it is observed that the proposed method had the highest performance on the three sub-benchmarks other than A3. The proposed method also had small standard deviations, ranging from a minimum of 0.0029 to a maximum of 0.0083. Through these results, we found that STOC had outstanding performance, robust to forecasting horizons. Fig. 5 also confirms this fact. In addition, to check whether there is a statistically significant difference between the performances of the best baseline and the proposed method, a one-sample t-test was performed for the null and alternative hypotheses in Eq. (11) by setting the best baseline of each sub-benchmark as μ_0. The t statistic and p-value are shown at the bottom of Table 5.

$$\begin{aligned} H_0 &: \mu = \mu_0 \\ H_1 &: \mu \neq \mu_0 \end{aligned} \tag{11}$$

The results of the t-test confirm that the proposed STOC has considerably lower p-values for all datasets. Thus, it can be verified that the

Table 4
Performance on Yahoo S5 benchmark.

Benchmark	A1		A2		A3		A4	
Method/Metric	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
LSTM-AD	0.8121	–	0.7348	–	0.5781	–	0.5811	–
DeepAnT	0.8976	0.4600	0.9614	0.9400	0.9284	0.8700	0.8597	0.6800
TadGAN	–	0.8000	–	0.8670	–	0.6850	–	0.6000
SISVAE	0.8760	0.4010	0.9810	0.7890	0.9930	**0.9510**	0.9240	**0.8200**
Informer-1	0.9133	0.7378	0.9791	0.7970	0.7209	0.4163	0.7653	0.3873
Anomaly Transformer	0.8963	0.7006	0.9960	0.9064	0.6874	0.2811	0.7211	0.2674
STOC-1	0.9226	0.7502	0.9982	0.9901	0.9848	0.9472	0.9321	0.7854
STOC-4	0.9388	0.7899	**1.0000**	**1.0000**	0.9903	0.9195	**0.9391**	0.7306
STOC-8	0.9410	0.7784	0.9998	0.9846	0.9889	0.8968	0.9255	0.6683
STOC-12	0.9334	0.7795	0.9996	0.9813	**0.9932**	0.8860	0.9214	0.6439
STOC-16	0.9284	0.8012	0.9979	0.9698	0.9917	0.8812	0.9172	0.6338
STOC-20	0.9347	0.8005	0.9900	0.9643	0.9927	0.9426	0.9377	0.7191
STOC-24	**0.9445**	**0.8014**	0.9993	0.9263	0.9911	0.9403	0.9327	0.7130

Table 5
AUROC comparison of the best performance of LSTM and CNN-based baselines and average performance of the proposed method on Yahoo S5 benchmark.

Benchmark		A1	A2	A3	A4
STOC	avg	0.9348	0.9978	0.9904	0.9294
	std	0.0075	0.0035	0.0029	0.0083
Best baseline		0.8976(D)	0.9810(S)	0.9930(S)	0.9240(S)
t statistic		13.1229	12.6996	−2.3720	1.7213
p-value		1.2080×10^{-5}	1.4618×10^{-5}	0.0554	1.1360

proposed STOC has improved from the LSTM or CNN-based methods on three benchmarks. In particular, the performance improvement for the A1 and A2 benchmarks is statistically significant at a significance level of 0.01.

The experimental results also show that the proposed method outperformed other Transformer-based methods for time series data, such as Informer and Anomaly Transformer. Compared with the Informer, the proposed method improved AUROC and F1-score by up to 37.% and 44.5%, respectively. In the case of Anomaly Transformer, the maximum improvement of AUROC and F1-score were 127.5% and 237.0%. The two Transformer-based models were comparable to or surpassed the LSTM or CNN-based models in A1 and A2 sub-benchmarks, but not in A3 and A4, where data distribution changes frequently. It shows that Transformer could work well with stable time series without distribution shifts, but its performance degenerated in a time series with distribution shifts. On the other hand, the proposed method showed better performance than LSTM, CNN-based models on all four sub-benchmarks. Compared to Informer and Anomaly Transformer, the proposed method, which fuses multi-level representations of time series, was more robust to distribution shifts. As Table 4 shows, the proposed method was found to be more potent than all baseline methods for detecting anomalies in time series regardless of distribution shift.

As shown in Table 6, for the NeurIPS-TS datasets, the proposed method achieved a higher AUROC than the two comparative methods in 70% (14 out of 20) of datasets. On the other hand, Informer and Anomaly Transformer ranked first in only 25% and 5% of datasets, respectively. In particular, STOC showed the best performance in more than 83% of datasets, including collective anomalies. This shows that STOC can detect continuous anomalies defined as a different pattern from normal data. However, for the point-global dataset, Informer-1, whose forecasting horizon is 1, showed better performance than STOC-1. However, compared with Informer, the proposed method has a better detecting ability for more complex anomalies, which requires understanding of the sophisticated context in time series. With respect to efficiency, STOC has about 9 million parameters, while Informer has 11 million parameters. Consequently, STOC is a better choice for covering various types of anomalies with less computation cost. In addition, we observed that the proposed method is improved by tuning the hyperparameter, τ. Among the candidate τ values (1, 4, 8, 12, 16, 20, 24), the best performance is presented as STOC-best in Table 6.

5.2. Qualitative results

This study aims to evaluate the anomaly detection performance of the proposed method not only quantitatively but also qualitatively. Thus, we visualized the predicted values, actual values, and anomaly scores computed by the proposed method for each sub-benchmark of the Yahoo S5 Benchmark. Figs. 6–9 display the actual values of the time-series data belonging to each sub-benchmark, the predicted values derived from the proposed method, and the computed anomaly scores.

Fig. 6 shows the results of the 27th time-series data of the A1 benchmark. The predicted values in the orange color closely imitate the global cycle and seasonality of the actual values in the blue color. Moreover, the predicted value at the timestamp where an actual anomaly occurred (indicated by the red vertical line) follows the distribution of normal data. Hence, the difference between the actual value and the predicted value at this timestamp is large. This large difference indicates a large anomaly score, as seen in the anomaly score plot at the bottom of Fig. 6. Therefore, anomalies can be detected using the threshold.

When the test data are inferred after learning the trend and cycle of normal data for the fourth time-series of the A2 benchmark, which is synthetic data, a large prediction error is observed in Fig. 7 for the rapidly increasing data point. Thus, a real anomaly was detected accurately at the corresponding timestamp. Furthermore, even if an actual anomaly occurred in the inference phase, the predicted value following normal data would be generated immediately afterward. From this phenomenon, it can be concluded that the model can predict normal values even if the input sequence contains anomalies during inference. Besides, the prediction error of actual anomalies and normal data points differ by 0.4 or more; hence, it is possible to perform robust anomaly detection to the threshold.

Fig. 8 shows the results of the experiment on the A3 benchmark. The training was carried out on the training data that contained many anomalies. Since the proposed method considers both the local and global information of the time-series, the prediction model can learn the distribution of normal data without being greatly affected by local anomalies. This way, the trend and seasonality of the normal time-series are learned. The prediction error was around 0.1 for normal data and over 0.3 for anomalies. Thus, the anomaly score can be used distinguish anomalies from normal data. Consequently, all anomalies in the test data were detected.

The experimental results on the A4 benchmark, which contains more than one change point, are shown in Fig. 9. Here, a change point refers to a point where the data distribution changes. The training data have two main behaviors; the test data, on the other hand, have a decreasing trend, which was not observed during training. In this manner, even for new data having a pattern with a different amplitude and trend from the training data, the model can produce predictions that follow the normal data distribution. Based on this result, it can

Table 6
Performance on NeurIPS-TS benchmark.

Benchmark	Point global (α = 0.05)		Point global (α = 0.10)		Point global (α = 0.15)		Point global (α = 0.20)	
Method/Metric	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
Informer-1	**0.9544**	0.8471	**0.9740**	**0.8662**	**0.9315**	**0.8235**	**0.9433**	**0.8208**
Anomaly Transformer	0.9689	0.8571	0.9186	0.6667	0.8730	0.6270	0.8929	0.6799
STOC-1	0.9527	**0.8864**	0.9301	0.8519	0.8912	0.8056	0.8836	0.7883
STOC-best	0.9877	0.9157	0.9691	0.8759	0.9359	0.8056	0.9485	0.8344
Benchmark	Point contextual (α = 0.05)		Point contextual (α = 0.10)		Point contextual (α = 0.15)		Point contextual (α = 0.20)	
Informer-1	0.7433	0.4646	0.7332	0.4056	0.8100	0.5813	0.7170	0.4444
Anomaly Transformer	0.7631	0.3158	0.6849	0.2920	0.7267	0.3687	0.7057	0.4305
STOC-1	**0.8916**	**0.6154**	**0.8160**	**0.5612**	**0.8677**	**0.6429**	**0.8221**	**0.6166**
STOC-best	0.9089	0.6329	0.8389	0.6815	0.8735	0.7446	0.8280	0.6531
Benchmark	Collective global (α = 0.05)		Collective global (α = 0.10)		Collective global (α = 0.15)		Collective global (α = 0.20)	
Informer-1	0.7257	0.6192	0.6523	0.7609	**0.6594**	**0.8949**	0.6716	0.9318
Anomaly Transformer	0.5919	0.6000	0.5947	0.7603	0.5614	0.8943	0.5560	0.9318
STOC-1	**0.7797**	**0.6701**	**0.8057**	**0.7940**	0.6486	**0.8949**	**0.7892**	**0.9324**
STOC-best	0.7969	0.6775	0.8057	0.7940	0.7602	0.8954	0.7892	0.9324
Benchmark	Collective seasonal (α = 0.05)		Collective seasonal (α = 0.10)		Collective seasonal (α = 0.15)		Collective seasonal (α = 0.20)	
Informer-1	0.7561	0.6675	0.6273	0.7984	0.5546	0.8743	0.6391	0.9324
Anomaly Transformer	0.6949	0.6289	0.5021	0.7989	0.7041	0.8748	**0.7590**	**0.9366**
STOC-1	**0.8235**	**0.7057**	**0.8214**	**0.8246**	**0.8535**	**0.8889**	0.7569	0.9324
STOC-best	0.8235	0.7057	0.8214	0.8246	0.8535	0.8889	0.7569	0.9324
Benchmark	Collective trend (α = 0.05)		Collective trend (α = 0.10)		Collective trend (α = 0.15)		Collective trend (α = 0.20)	
Informer-1	0.1468	**0.5760**	0.1858	**0.8127**	0.4037	**0.8259**	0.2394	**0.9267**
Anomaly Transformer	0.1536	0.5737	0.2147	**0.8127**	0.4120	0.8243	0.2164	**0.9267**
STOC-1	**0.1689**	0.5737	**0.2355**	**0.8127**	**0.8677**	0.6429	**0.8221**	0.6166
STOC-best	0.1689	0.5737	0.2670	0.8127	0.8735	0.7746	0.8622	0.6515

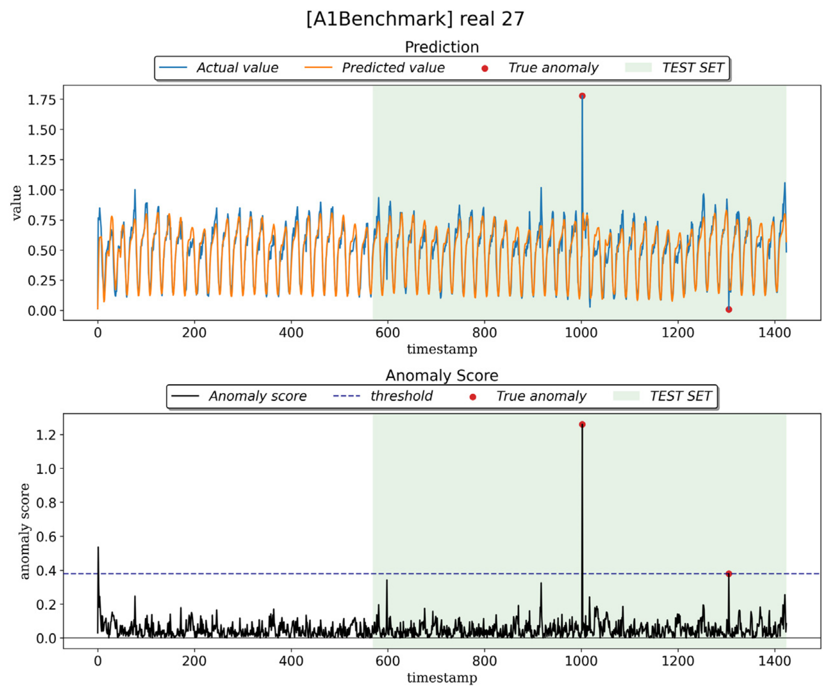


Fig. 6. Predictions and anomaly scores of a sample from A1 benchmark.

be interpreted that Transformer had sufficient capacity to learn the varying behaviors in the given time-series.

As shown above, through qualitative evaluation, we confirm that the prediction and the anomaly score calculation method proposed in this study effectively detect anomalies regardless of the characteristic of the given data. However, the experimental results for the 22nd data of the A4 benchmark in Fig. 10 show that although the anomaly score of an actual anomaly is clearly distinguishable from the surrounding normal data, it can be challenging to detect this anomaly. Because when the amplitude of the given time-series varies greatly, the average prediction error differs too. This result reveals a limitation that not all anomalous data can be detected using a static threshold.

6. Ablation study

We explored the effect of each component of STOC on the Yahoo S5 benchmark. We added two components to the standard Transformer: the stacked representation of Transformer encoder layers and the 1d-CNN decoder. In all cases, τ is 1. Table 7 presents the average performance of ablations on each sub-benchmark. Regardless of decoder, the stacked encoder representation improved performance

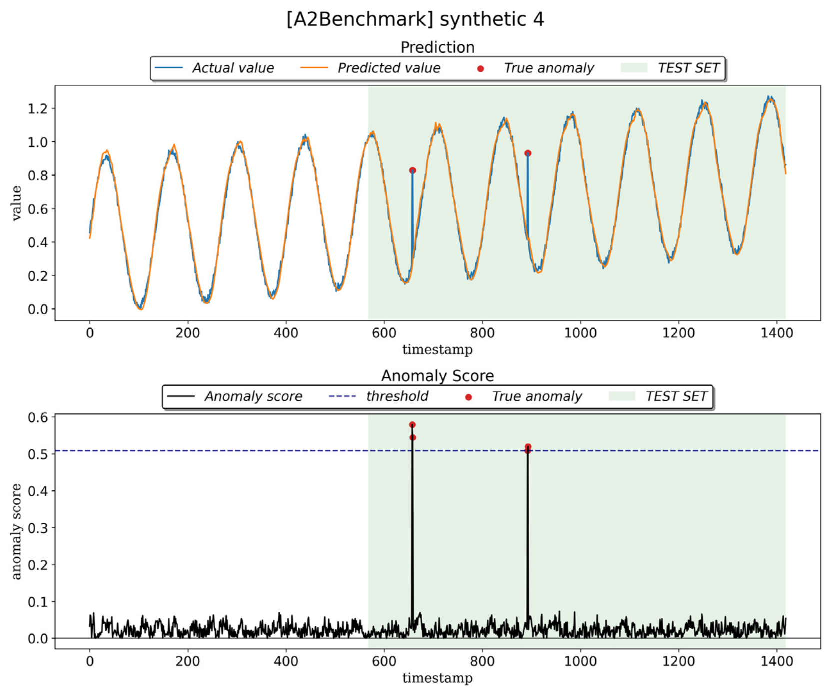


Fig. 7. Predictions and anomaly scores of a sample from A2 benchmark.

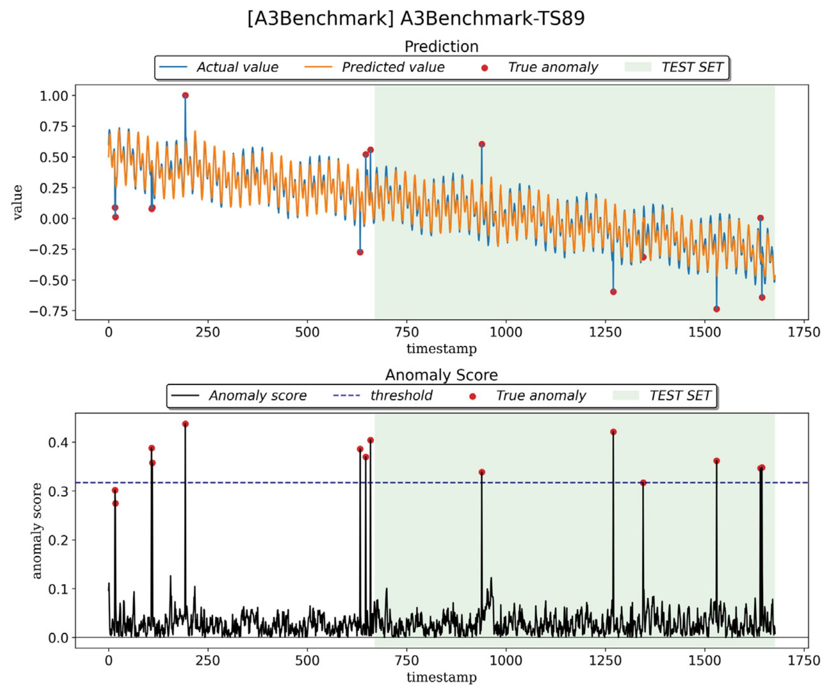


Fig. 8. Predictions and anomaly scores of a sample from A3 benchmark.

compared to the standard Transformer by 1.3% and 22% in terms of AUROC and F1-score, respectively. This means that integrating multi-level representations in Transformer encoder rather than using a single top-layer representation can yield richer features of input time series. When equally using the stacked representation from an encoder, the standard Transformer decoder outperformed the 1d-CNN decoder on A1, while 1d-CNN beat the Transformer decoder on A3 and A4. Note that A3 and A4 include anomalies in the training dataset, and the distributions shift several times. Under this circumstance, 1d-CNN makes the forecasting model ignore the abnormal pattern in time windows by extracting only major normal features from the windows. This helps the proposed method capture normal patterns in time series, even with anomalous points like A3 or A4. In addition, the detection model must rapidly adapt to distribution changes in a real-time framework. From this perspective, 1d-CNN decoder is more powerful than the standard Transformer decoder for detecting anomalies in a challenging environment. Fig. 11 shows that STOC using both stacked encoder representation and 1d-CNN decoder has the advantage of robustly forecasting normal time series well, regardless of abnormal points in input time windows. Besides, since 1d-CNN decoder is smaller than

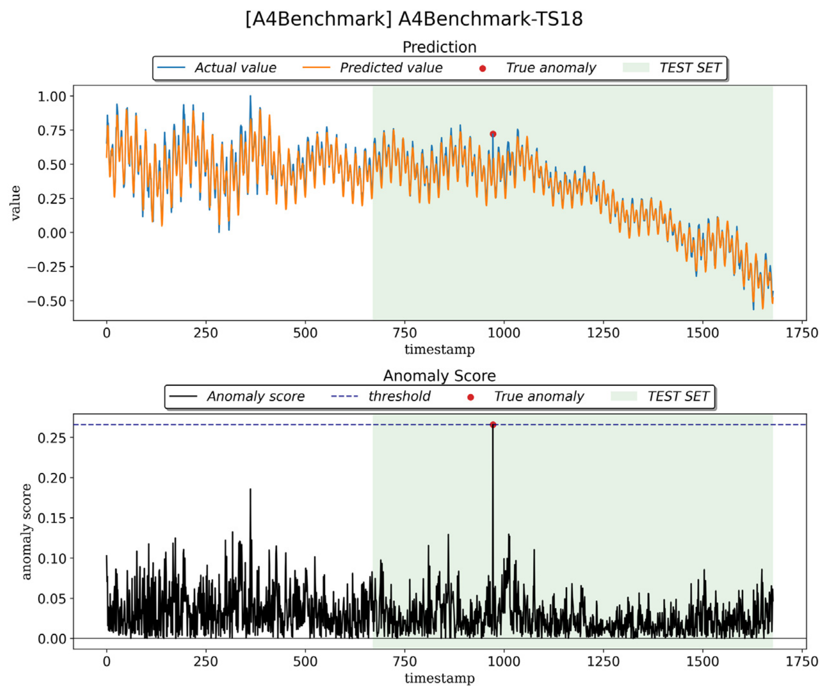


Fig. 9. Predictions and anomaly scores of a sample from A4 benchmark.

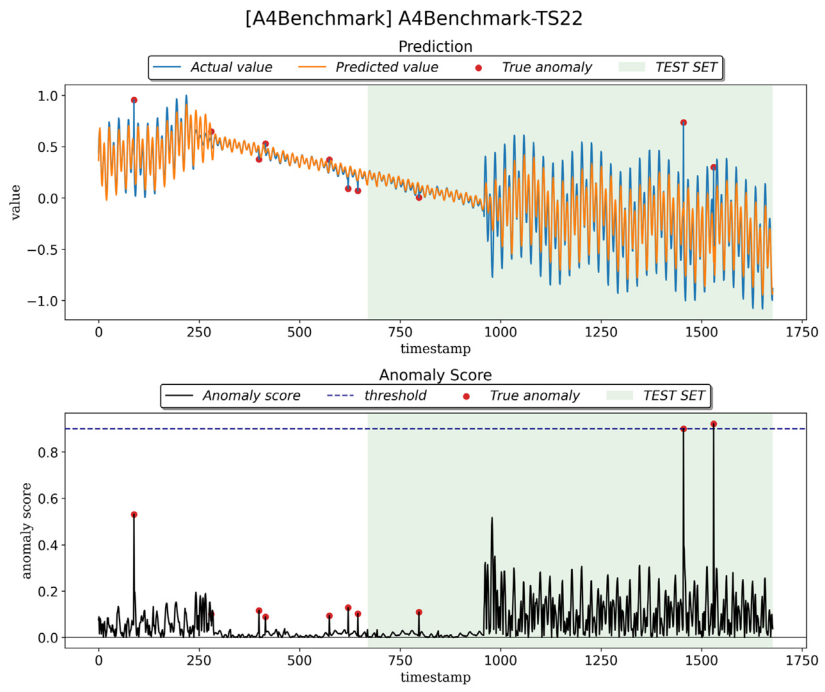


Fig. 10. Predictions and anomaly scores of a sample from A4 benchmark.

the standard Transformer decoder in size (Table 8), the proposed method succeeded in achieving both effectiveness of multi-level rich information and training efficiency.

7. Conclusion

Time-series anomaly detection prevents problems in advance by recognizing potential failures based on the time-series data generated by the system. Therefore, time-series anomaly detection has been used for system maintenance in manufacturing, IT, healthcare, and so on. Since highly accurate anomaly detection prevents costs incurred due to unnecessary system interruption, it can be said that the anomaly detection performance positively correlates with the system's efficiency. However, it is challenging to collect labels for the anomalies because anomalies do not occur frequently in the real world. Thus, studies have been conducted mainly based on unsupervised learning. Most studies use RNNs or CNNs to capture the continuous structure of the time-series.

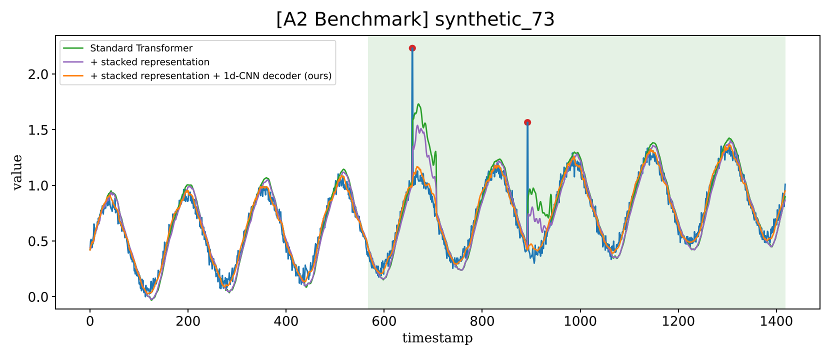


Fig. 11. Prediction results of STOC and its ablations on Yahoo S5 A2 benchmark.

Table 7
Ablation study of various components of STOC on Yahoo S5 benchmark.

Benchmark	A1		A2		A3		A4	
Method/Metric	AUROC	F1	AUROC	F1	AUROC	F1	AUROC	F1
Standard Transformer encoder + Standard Transformer decoder	0.9105	0.7459	<u>0.9988</u>	0.9409	0.9785	0.8096	0.9195	0.6426
Stacked Transformer encoder representation + Standard Transformer decoder	**<u>0.9271</u>**	**<u>0.7668</u>**	**<u>0.9998</u>**	<u>0.9788</u>	<u>0.9839</u>	<u>0.8410</u>	<u>0.9242</u>	<u>0.6680</u>
Stacked Transformer encoder representation + 1d-CNN decoder (ours)	<u>0.9226</u>	<u>0.7502</u>	0.9982	**<u>0.9901</u>**	**<u>0.9848</u>**	**<u>0.9472</u>**	**<u>0.9321</u>**	**<u>0.7854</u>**

Table 8
The size of STOC and its ablations.

Method	# of parameters
Standard Transformer encoder + Standard Transformer decoder	8,418,048
Stacked Transformer encoder representation + Standard Transformer decoder	11,770,368
Stacked Transformer encoder representation + 1d-CNN decoder (ours)	9,006,080

In this study, we proposed a prediction-based unsupervised time-series anomaly detection method using Transformer because Transformer has shown excellent performance in capturing dependencies within a sequence. The predicted values were calculated through a prediction model that learned the distribution of the normal training data, whereby data that did not follow this distribution were considered anomalous. The prediction model consisted of an encoder that accumulates the hierarchical information from the sequence modeled by the Transformer layers and a decoder that utilizes a 1D convolution to combine multi-level information effectively. The proposed model derived feature representations from the encoder, considering both the global trend information and the local variability of the given time-series. In the decoder, we extracted the common information from the feature representation received from the encoder via the 1D convolution operation and predicted the future values using these extracted representations. This process allowed the prediction model to learn the distribution of normal data successfully. The experimental results showed that the proposed method has superior performance to the baselines on the benchmark datasets. In addition, the ablation study confirmed the effectiveness of stacking transformer encoder representations and using 1d-CNN as a decoder.

Furthermore, we designed this method to be applicable not only to univariate time-series data but also to multivariate time-series data. Accordingly, the multivariate aspect can be explored in future research. In addition, it is expected that better detection performance can be achieved by applying dynamic thresholding, which can handle the difference in the amplitudes of the time-series.

CRediT authorship contribution statement

Jina Kim: Methodology, Software, Writing – original draft, Writing – review & editing. **Hyeongwon Kang:** Software, Validation, Investigation. **Pilsung Kang:** Conceptualization, Supervision, Project administration, Funding acquisition.

Data availability

Yahoo S5 Benchmark datasets can be downloaded upon request from the official site. NeurIPS-TS can be downloaded from the author's github page.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2022R1A2C2005455) and the Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2021-0-00034, Clustering technologies of fragmented data for time-based data analysis).

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Braei, M., Wagner, S., 2020. Anomaly detection in univariate time-series: A survey on the state-of-the-art. arXiv:2004.00433.

Breunig, M.M., Kriegel, H.-P., Ng, R.T., Sander, J., 2000. LOF: Identifying Density-Based Local Outliers. Association for Computing Machinery, New York, NY, USA, pp. 93–104. http://dx.doi.org/10.1145/342009.335388.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., Houlsby, N., 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In: International Conference on Learning Representations. URL: https://openreview.net/forum?id=YicbFdNTTy.

Fang, M., Damer, N., Boutros, F., Kirchbuchner, F., Kuijper, A., 2020. Deep learning multi-layer fusion for an accurate iris presentation attack detection. In: 2020 IEEE 23rd International Conference on Information Fusion (FUSION). pp. 1–8. http://dx.doi.org/10.23919/FUSION45008.2020.9190424.

Gao, J., Song, X., Wen, Q., Wang, P., Sun, L., Xu, H., 2020. RobustTAD: Robust time series anomaly detection via decomposition and convolutional neural networks. arXiv:2002.09545.

Geiger, A., Liu, D., Alnegheimish, S., Cuesta-Infante, A., Veeramachaneni, K., 2020. TadGAN: Time series anomaly detection using generative adversarial networks. In: 2020 IEEE International Conference on Big Data (Big Data). pp. 33–43. http://dx.doi.org/10.1109/BigData50022.2020.9378139.

Gulati, A., Qin, J., Chiu, C.-C., Parmar, N., Zhang, Y., Yu, J., Han, W., Wang, S., Zhang, Z., Wu, Y., Pang, R., 2020. Conformer: Convolution-augmented transformer for speech recognition. In: Interspeech 2020. ISCA, http://dx.doi.org/10.21437/interspeech.2020-3015.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Comput. 9 (8), 1735–1780. http://dx.doi.org/10.1162/neco.1997.9.8.1735.

Huang, C.-Z.A., Vaswani, A., Uszkoreit, J., Shazeer, N., Simon, I., Hawthorne, C., Dai, A.M., Hoffman, M.D., Dinculescu, M., Eck, D., 2018. Music transformer. http://dx.doi.org/10.48550/ARXIV.1809.04281, URL: https://arxiv.org/abs/1809.04281.

Lai, K.-H., Zha, D., Xu, J., Zhao, Y., Wang, G., Hu, X., 2021. Revisiting time series outlier detection: Definitions and benchmarks. In: Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1). URL: https://openreview.net/forum?id=r8IvOsnHchr.

Laptev, N., Amizadeh, S., Flint, I., 2015. Generic and scalable framework for automated time-series anomaly detection. In: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. KDD '15, Association for Computing Machinery, New York, NY, USA, pp. 1939–1947. http://dx.doi.org/10.1145/2783258.2788611.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., Yan, X., 2019. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In: Wallach, H., Larochelle, H., Beygelzimer, A., d' Alché-Buc, F., Fox, E., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Vol. 32. Curran Associates, Inc..

Li, L., Yan, J., Wang, H., Jin, Y., 2021. Anomaly detection of time series with smoothness-inducing sequential variational auto-encoder. IEEE Trans. Neural Netw. Learn. Syst. 32 (3), 1177–1191. http://dx.doi.org/10.1109/TNNLS.2020.2980749.

Liu, F.T., Ting, K.M., Zhou, Z.-H., 2008. Isolation forest. In: 2008 Eighth IEEE International Conference on Data Mining. pp. 413–422. http://dx.doi.org/10.1109/ICDM.2008.17.

Loshchilov, I., Hutter, F., 2019. Decoupled weight decay regularization. In: International Conference on Learning Representations. URL: https://openreview.net/forum?id=Bkg6RiCqY7.

Ma, C., Mu, X., Sha, D., 2019. Multi-layers feature fusion of convolutional neural network for scene classification of remote sensing. IEEE Access 7, 121685–121694. http://dx.doi.org/10.1109/ACCESS.2019.2936215.

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

Malhotra, P., Vig, L., Shroff, G., Agarwal, P., et al., 2015. Long short term memory networks for anomaly detection in time series. In: Proceedings, Vol. 89. pp. 89–94.

Marjani, M., Nasaruddin, F., Gani, A., Karim, A., Hashem, I.A.T., Siddiqa, A., Yaqoob, I., 2017. Big IoT data analytics: Architecture, opportunities, and open research challenges. IEEE Access 5, 5247–5261. http://dx.doi.org/10.1109/ACCESS.2017.2689040.

Martí, L., Sanchez-Pi, N., Molina, J.M., Garcia, A.C.B., 2015. Anomaly detection based on sensor data in petroleum industry applications. Sensors 15 (2), 2774–2797.

Munir, M., Siddiqui, S.A., Dengel, A., Ahmed, S., 2018. DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. Ieee Access 7, 1991–2005.

Niu, Z., Yu, K., Wu, X., 2020. LSTM-based VAE-GAN for time-series anomaly detection. Sensors 20 (13).

Schölkopf, B., Platt, J.C., Shawe-Taylor, J., Smola, A.J., Williamson, R.C., 2001. Estimating the support of a high-dimensional distribution. Neural Comput. 13 (7), 1443–1471.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I., 2017. Attention is all you need. In: Guyon, I., Luxburg, U.V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., Garnett, R. (Eds.), Advances in Neural Information Processing Systems, Vol. 30. Curran Associates, Inc..

Wang, Q., Li, F., Xiao, T., Li, Y., Li, Y., Zhu, J., 2018. Multi-layer representation fusion for neural machine translation. In: Proceedings of the 27th International Conference on Computational Linguistics. Association for Computational Linguistics, Santa Fe, New Mexico, USA, pp. 3015–3026, URL: https://aclanthology.org/C18-1255.

Wen, Q., Sun, L., Yang, F., Song, X., Gao, J., Wang, X., Xu, H., 2021. Time series data augmentation for deep learning: A survey. In: Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence. International Joint Conferences on Artificial Intelligence Organization, http://dx.doi.org/10.24963/ijcai.2021/631.

Wu, N., Green, B., Ben, X., O'Banion, S., 2020. Deep transformer models for time series forecasting: The influenza prevalence case. arXiv:2001.08317.

Wu, H., Xu, J., Wang, J., Long, M., 2021. Autoformer: Decomposition transformers with auto-correlation for long-term series forecasting. In: Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., Vaughan, J.W. (Eds.), Advances in Neural Information Processing Systems, Vol. 34. Curran Associates, Inc., pp. 22419–22430.

Xu, L.D., He, W., Li, S., 2014. Internet of things in industries: A survey. IEEE Trans. Ind. Inform. 10 (4), 2233–2243. http://dx.doi.org/10.1109/TII.2014.2300753.

Xu, J., Wu, H., Wang, J., Long, M., 2022. Anomaly transformer: Time series anomaly detection with association discrepancy. In: International Conference on Learning Representations. URL: https://openreview.net/forum?id=LzQQ89U1qm_.

Zhou, H., Zhang, S., Peng, J., Zhang, S., Li, J., Xiong, H., Zhang, W., 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35. pp. 11106–11115.