

Article

Multivariate Time Series Anomaly Detection Using Working Memory Connections in Bi-Directional Long Short-Term Memory Autoencoder Network

Xianghua Ding , Jingnan Wang, Yiqi Liu and Uk Jung * 

Department of Business Administration, School of Business, Dongguk University-Seoul, Seoul 04620, Republic of Korea; 2022126611@dgu.ac.kr (X.D.); layla@dgu.ac.kr (J.W.); jtankjlyq@dongguk.edu (Y.L.)

* Correspondence: ukjung@dongguk.edu

Abstract: “Normal” events are characterized as data patterns or behaviors that align with expected operational conditions, while “anomalies” are defined as deviations from these patterns, potentially signaling faults, errors, or unexpected system behaviors. The timely and accurate detection of anomalies plays a critical role in domains such as industrial manufacturing, financial transactions, and other related domains. In the context of Industry 4.0, the proliferation of sensors has resulted in a massive influx of time series data, making the anomaly detection of such multivariate time series data a popular research area. Long Short-Term Memory (LSTM) has been extensively recognized as an effective framework for modeling and processing time series data. Previous studies have combined Bi-directional Long Short-Term Memory (Bi-LSTM) architecture with Autoencoder (AE) for multivariate time series anomaly detection. However, due to the inherent limitations of LSTM, Bi-LSTM-AE still cannot overcome these drawbacks. Our study replaces the LSTM units within the Bi-LSTM-AE architecture of existing research with Working Memory Connections for LSTM units and demonstrates that this architecture performs better in the field of multivariate time series anomaly detection compared to using standard LSTM units. The model we proposed not only outperforms the baseline models but also demonstrates greater robustness across various scenarios.

Keywords: anomaly detection; long short-term memory; time series data; autoencoder; multivariate time series



Academic Editors: Yutaka Ishibashi and Alessandro Lo Schiavo

Received: 16 January 2025

Revised: 1 February 2025

Accepted: 3 March 2025

Published: 6 March 2025

Citation: Ding, X.; Wang, J.; Liu, Y.; Jung, U. Multivariate Time Series Anomaly Detection Using Working Memory Connections in Bi-Directional Long Short-Term Memory Autoencoder Network. *Appl. Sci.* **2025**, *15*, 2861. <https://doi.org/10.3390/app15052861>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Anomalies are patterns in data that do not conform to a well defined notion of normal behavior [1]. Anomaly detection involves identifying rare events, items, or observations that significantly differ from expected behaviors or established patterns [2]. Anomaly detection is widely applied across numerous domains, including network monitoring, healthcare, smart devices, smart cities, the Internet of Things, fraud detection, cloud computing, and beyond [3]. In real-world applications, data reported by sensors are often in the form of multivariate time series. Such data may exhibit correlations among different variables as well as autocorrelations across individual observations. Accurately identifying anomalies within such complex datasets is a challenging task. With advancements in artificial intelligence, deep learning and related techniques have been widely adopted in the field of anomaly detection, providing a variety of effective solutions for detecting anomalies in multivariate time series data.

Traditional methods for anomaly detection predominantly rely on statistical measures and data density. The earliest work in this domain was introduced by Shewhart, who proposed algorithms for detecting anomalies in process quality control applications [4]. Subsequently, more advanced statistical techniques, such as Grubbs's test [5], student's *t*-test [6], Hotelling's *t*-squared test [7], and chi-squared test [8] were developed to enhance anomaly detection capabilities. The core principle of statistical-based anomaly detection is that anomalies tend to appear in low-probability regions, whereas normal data instances are typically located in high-probability regions. These algorithms are among the earliest approaches proposed for anomaly detection. Their methodology involves first fitting a statistical model to a given dataset, followed by performing statistical tests on new data to evaluate its conformity to the model. Data points consistent with the model are classified as normal, while those that deviate are identified as anomalies [9]. However, statistical-based methods are limited by the assumption that the data conforms to a specific distribution, such as a Gaussian distribution. When this assumption is not met, their effectiveness diminishes significantly. Furthermore, these methods struggle with high-dimensional data, as conventional statistical approaches are inherently unsuitable for such scenarios. Although dimensionality reduction techniques like Principal Component Analysis (PCA) can mitigate this issue, they are often inadequate for high-dimensional data with nonlinear relationships. These limitations restrict the applicability of statistical-based methods in complex anomaly detection tasks.

With the advent of Industry 4.0, researchers and practitioners face an explosion in the volume of data that needs to be processed, often accompanied by nonlinear relations between variables in large datasets. Concurrently, rapid advancements in computational hardware have alleviated the computational complexity constraints, allowing data-driven approaches to become mainstream. Among these, machine learning-based anomaly detection has emerged as a prominent research area. Various machine learning algorithms implement anomaly detection through different methodologies. The distance-based algorithm *k*-Nearest Neighbors (*k*-NNs) [10] operates on the core idea that an anomalous data point is far from its *k*-NN, with the anomaly score computed as the distance between the data instance and its *k*-NN. Local Outlier Factor (LOF) [11] is another well-known and widely used anomaly detection algorithm, which evaluates the relative density of a data point with respect to its *k*-Nearest Neighborhood. Mika et al. [12] combined statistical foundations with machine learning kernel methods to propose Kernel Principal Component Analysis (KPCA), addressing the limitations of PCA in handling high-dimensional data with nonlinear relations. Sheridan et al. [13] successfully applied a clustering-based algorithm, utilizing density-based spatial clustering of applications with noise (DBSCAN), for flight anomaly detection during the aircraft approach phase. Additionally, isolation-based algorithms, subspace-based algorithms, and others have also been proven effective in the domain of anomaly detection. However, while these machine learning methods have somewhat addressed the problem of anomaly detection in multivariate data with nonlinear relations, they still exhibit varying degrees of failure when confronted with higher-dimensional data and data with autocorrelation.

With the development of artificial intelligence and neural networks, deep learning-based anomaly detection has gradually become mainstream [14]. This approach can handle massive amounts of data and has relatively lenient requirements for training data. Depending on the labels available in the dataset, one can choose unsupervised, semi-supervised, or supervised learning [1]. Among neural network-based methods, Autoencoder (AE)-based approaches are predominantly used due to their powerful capability in advanced data representation. These methods first encode and compress the input into latent variables and then decode it back to a state close to the original dataset, ensuring the AE learns only

the essential information in the data [15]. Since AE lacks specific targets, they adopt an unsupervised training approach to learn data representation [16]. However, the massive influx of high-dimensional time series data generated in Internet of Things (IoT) and smart manufacturing environments poses challenges that conventional AE cannot address due to the inherent time series properties.

Inspired by Cho [17], researchers have begun integrating Recurrent Neural Networks (RNNs) with AE, proposing a series of hybrid models. By leveraging RNNs' ability to remember past information, these models aim to handle multivariate time series data. To mitigate the gradient vanishing and exploding issues in RNNs, Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are used as substitutes. Experiments in skeleton-based gait anomaly recognition have demonstrated the capabilities of these models, with Long Short-Term Memory Autoencoder (LSTM-AE) combinations outperforming Gated Recurrent Unit Autoencoder (GRU-AE) [18]. LSTM-AE is widely used as a foundational approach for time series anomaly detection and has spawned numerous variants. Park et al. [19] proposed a Variational Autoencoder (VAE) based on LSTM, successfully detecting anomalies in robot-assisted feeding with superior performance to baseline models. Lee et al. [16] applied Bi-directional Long Short-Term Memory (Bi-LSTM) networks to AE, where one unidirectional LSTM processes data from past to future and the other from future to past, enhancing anomaly detection accuracy in smart metering systems. Raihan and Ahmed [20] demonstrated that Bi-LSTM-AE outperformed conventional LSTM-AE on wind power datasets. However, these studies all build on LSTM, inheriting its inherent properties.

Gers and Schmidhuber [21] argued that due to the gate connectivity approach of conventional LSTMs, this can lead to loss of necessary information and a decline in network performance. Landi et al. [22] proposed Working Memory Connections (WMCs) for LSTM, providing an efficient way to utilize intra-cell knowledge within the network. This design significantly outperformed vanilla LSTM and addressed key issues in previous formulations (LSTM with peephole connections). It has been validated in two different toy problems, as well as in tasks such as digit recognition, language modeling, and image captioning. However, there has been no example of its use in the field of anomaly detection. We believe this variant can enhance existing LSTM-based anomaly detection algorithms by allowing memory cell state to influence gates, thereby improving information flow control and making anomaly detection algorithms more sensitive.

Recently, several studies have introduced time series anomaly detection methods that do not rely on LSTM. Jeong et al. [23] proposed a method named AnomalyBERT, a self-supervised Transformer-based approach for time series anomaly detection, which leverages a novel data degradation scheme with synthetic outliers to effectively capture temporal contexts and inter-variable relationships. Yang et al. [24] proposed a method named DCdetector, a dual-attention contrastive representation learning algorithm for time series anomaly detection, which enhances representation differences between normal and anomalous points without relying on reconstruction error. Xu et al. [25] proposed a calibrated one-class classification method for unsupervised time series anomaly detection (COUTA), leveraging Temporal Convolutional Networks and dual calibration techniques to achieve contamination-tolerant, anomaly-informed normality learning with state-of-the-art performance. However, it is worth noting that no universally optimal model or approach currently exists in the academic community.

In this paper, we propose an anomaly detection model called Bi-directional Long Short-Term Memory with Working Memory Connections-Autoencoder (Bi-LSTM-WMCs-AE). It applies Bi-LSTM with WMCs to an AE network. While the Bi-LSTM with the WMCs network effectively learns the time-related information inherent in the multivariate timeseries dataset, the AE part acquires the ability to reconstruct normal data by learning

from normal data. The difference between the input data and the output data, known as the reconstruction error, is used for anomaly detection.

The main contributions of our study are as follows:

- (1) We propose a novel anomaly detection model for multivariate time series data. Building on the widely adopted Bi-LSTM-AE framework, we introduce a variant of LSTM, referred to as Working Memory Connections for LSTM, to replace the traditional LSTM units. This variant enables the memory cell to influence the gate values, thereby facilitating more effective processing of time series data. To the best of our knowledge, our study is the first to apply this novel LSTM variant to the field of anomaly detection.
- (2) Our model leverages a reconstruction-based framework, making it inherently suited for semi-supervised learning. Unlike approaches that rely on extensive labeled anomaly data, our method requires only normal data for training, significantly reducing the dependency on scarce and resource-intensive expert-labeled anomaly datasets. This advantage addresses a critical challenge in real-world applications, where labeled anomaly data are often limited or unavailable.
- (3) We evaluated the performance of our model on diverse datasets that include both continuous sensor data and mixed data containing sensor and actuator reports. Our model demonstrated consistently strong performance across these scenarios, highlighting its robustness and scalability in handling various types of multivariate time series data.

The rest of this paper unfolds as follows. Section 2 discusses the related work. Section 3 describes the proposed model for multivariate time series anomaly detection. Section 4 presents a case study using the proposed model. Section 5 consists of our concluding remarks.

2. Related Work

2.1. AE

AE is a neural network architecture composed of stacked layers, often employing nonlinear activation functions to model complex, nonlinear relationships within the data. With this capability, AE is extensively employed for dimensionality reduction and feature extraction in high-dimensional datasets, providing a foundation for subsequent machine learning tasks. A standard AE primarily consists of three components: an encoder, a latent representation, and a decoder. The encoder maps high-dimensional input data into a more compact, lower-dimensional representation known as the latent representation. Subsequently, the decoder utilizes this latent representation to reconstruct the original input while minimizing information loss. The information loss between the original input and the reconstructed output is referred to as the reconstruction error. Figure 1 illustrates a standard AE model.

The encoder is responsible for mapping the original input data x , typically represented as an m -dimensional vector ($x \in \mathbb{R}^m$), into a lower-dimensional latent representation, often denoted as z . The encoding process can be mathematically expressed as follows:

$$z = g(W_e x + b_e) \quad (1)$$

where W_e is the weight matrix with dimensions $p \times m$, b_e is the bias vector with dimensions p , and g is an activation function. The dimensionality of the latent space z is specified as p , where p is a positive integer, representing the reduced dimensionality of the input data.

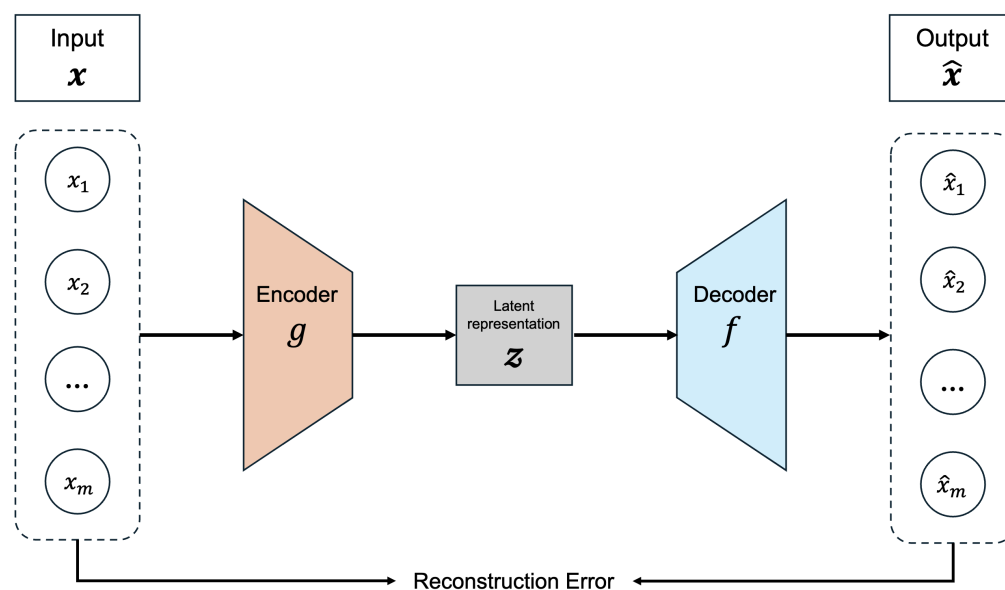


Figure 1. A simple Autoencoder architecture.

However, recent research has challenged the necessity of a bottleneck in anomaly detection models. Yong and Brintrup [26] have demonstrated that the latent space dimensionality does not necessarily need to be smaller than the dimensionality of the input data. Specifically, removing the bottleneck by either expanding the latent space or introducing skip connections can lead to improved performance in anomaly detection tasks. This finding suggests that non-bottlenecked autoencoders, which allow the model to learn more complex representations, may outperform their bottlenecked counterparts in certain contexts, particularly in tasks where a higher expressivity of the model is beneficial for detecting anomalies. Our study will also employ this overcomplete architecture Autoencoder in the subsequent sections.

The decoder utilizes the latent representation z to reconstruct the original input as accurately as possible. The reconstructed output is denoted by $\hat{x}$, while the difference between $\hat{x}$ and the original input x is referred to as the reconstruction error. Equation (2) formalizes the decoding process, where f is the activation function, W_d is the weight matrix with dimensions $m \times p$, b_d represents the bias vector with dimensions m , and $\hat{x}$ denotes the reconstructed input data. Equation (3) describes the computation of the reconstruction error, where L represents the reconstruction error, and m denotes the dimensionality of the original data, corresponding to the number of variables.

$$\hat{x} = f(W_d z + b_d) \quad (2)$$

$$L(x, \hat{x}) = \frac{1}{m} \sum_{i=1}^m (\hat{x}_i - x_i)^2 \quad (3)$$

2.2. LSTM

LSTM, an enhanced variant of RNN, effectively mitigates the vanishing gradient problem by adaptively regulating the input and output values of its gates, thereby controlling the retention or forgetting of information within the cell state [27]. This capability enables LSTM to more effectively capture and leverage temporal dependencies in time series data. The LSTM architecture consists of three primary components: the forget gate, the input gate, and the output gate, with the information flow between these components governed by the following equations:

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + b_f) \quad (4)$$

$$\tilde{c}_t = \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) \quad (5)$$

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + b_i) \quad (6)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \quad (7)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + b_o) \quad (8)$$

$$h_t = o_t \odot \tanh(c_t) \quad (9)$$

where f_t , i_t , and o_t represent the outputs of the forget gate, input gate, and output gate, respectively. These outputs are n -dimensional vectors, where n corresponds to the number of hidden units in the LSTM's hidden state. The sigmoid activation function constrains each element of the output vector to the range $[0,1]$, thereby facilitating the regulation of information flow. The hyperbolic tangent activation function maps each element of the output vector to the range $[-1,1]$, enabling the representation of both positive and negative values. This characteristic facilitates the modeling of more complex relationships and ensures a balanced gradient flow during training. W_{fx} , W_{cx} , W_{ix} , and W_{ox} represent the weight matrices between the input at the current time step x_t and the forget gate, input gate, and output gate, respectively. Each of these weight matrices has dimensions $n \times m$, where m is the dimensionality of the input vector. W_{fh} , W_{ch} , W_{ih} , and W_{oh} represent the weight matrices between the hidden state at the previous time step h_{t-1} and the forget gate, cell gate, input gate, and output gate, respectively. Each of these weight matrices has dimensions $n \times n$. b_f , b_c , b_i , and b_o are the n -dimensional bias vectors for the forget gate, input gate, and output gate, respectively.

2.3. Bi-LSTM

As an advanced extension of RNN, LSTM is highly effective at capturing temporal dependencies in time series data. However, it relies exclusively on past memory, which can lead to errors in prediction or classification tasks [28]. Bi-LSTM, on the other hand, leverages bidirectional information flow by incorporating both past-to-future and future-to-past dependencies, thereby enabling more accurate analysis and processing of time series data. This advantage has led to the widespread application of Bi-LSTM in existing studies on time series and contextual analysis.

2.4. LSTM-WMCs

LSTM-WMCs is a variant of LSTM that enables the memory cell to influence the value of the gates through a set of recurrent weights [22]. The core concept of this variant is that, in traditional LSTM units, the regulation of information flow through the three gates can be likened to the operation of three valves controlling the flow of information in fixed directions. The opening and closing of these valves are determined by x_t , which represents the current input, and h_{t-1} , which encapsulates past information. However, as indicated by Equation (9), h_t is, in fact, a subset of c_t . This implies that c_t retains certain valuable and significant information from previous time steps that is not captured in h_t . In the LSTM-WMCs variant, the cell state c_t is integrated into the gating mechanism, allowing

for more precise regulation of information flow within the gates. Figure 2 illustrates the structure of this LSTM variant.

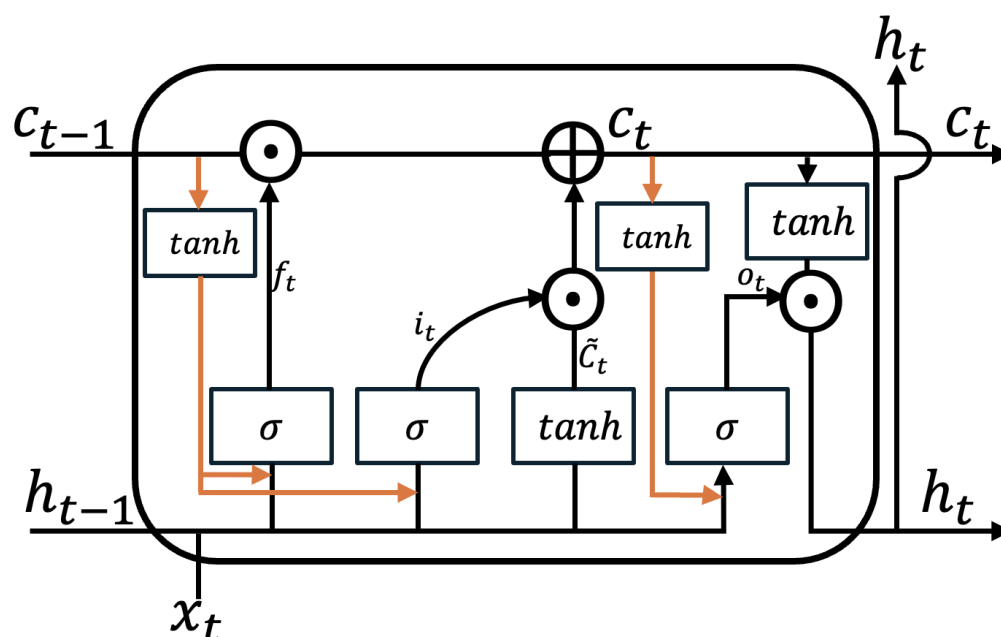


Figure 2. A working memory connection for LSTM structure.

The equations below highlight the modifications introduced in this novel LSTM variant compared to the traditional LSTM:

$$i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + \tanh(W_{ic}c_{t-1}) + b_i) \quad (10)$$

$$f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + \tanh(W_{fc}c_{t-1}) + b_f) \quad (11)$$

$$o_t = \sigma(W_{ox}x_t + W_{oh}h_{t-1} + \tanh(W_{oc}c_t) + b_o) \quad (12)$$

where W_{ic} , W_{fc} , and W_{oc} represent $n \times n$ weight matrices that define the connections between the cell state and the input gate, forget gate, and output gate, respectively. The hyperbolic tangent function here serves the same purpose as in Equation (9), ensuring that only relevant and appropriately scaled information is propagated between gates. This mechanism prevents the unbounded cell state from exerting excessive influence on the gate dynamics, thereby avoiding undesired saturation.

3. Proposed Method

In this section, we introduce our proposed model that uses the combination of Bi-LSTM-WMCs and AE to detect anomalies based on the analysis of multivariate time series data. We first provide the overview of our proposed model based on the three steps of data preprocessing, the Bi-LSTM-WMCs-AE model, and anomaly detection. We also provide a detailed description of the algorithm our model uses in terms of training and testing phases.

Step 1 : Data Preprocessing

Similar to models based on traditional LSTM, our proposed model requires transforming the original input data into a format suitable for processing by LSTM units, specifically multiple time windows. Typically, a complete multivariate time series is represented as a 2D array comprising n m -dimensional vectors, where n represents the total number of time points, and m denotes the number of variables in the time series. Since LSTM models

process data in multiple time windows, each represented as a 2D array, the number of time points in a single window, referred to as the window size (t), determines the time steps the model considers for decision-making.

To preprocess the data, a sliding window approach with a fixed size ($t \times m$) is applied, shifting the window by one time point at each step. This process transforms the original 2D array into $(n - t + 1)$ stacked 2D arrays, each with dimensions $(t \times m)$, representing the sequence of time windows. Figure 3 illustrates the details of the data preprocessing workflow.

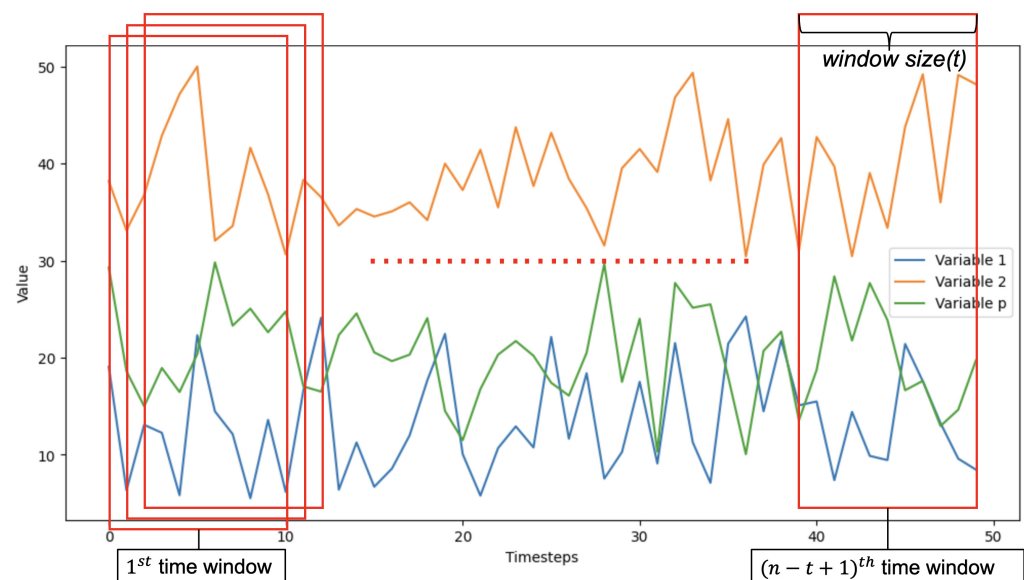


Figure 3. Example of data pre-processing divided into time windows. The red squares indicate the sliding windows applied during the preprocessing step.

Step 2: Online Training

In this step, the traditional LSTM units in the Bi-LSTM-AE network, a model widely employed in existing studies, are replaced with bidirectional LSTM-WMC units to enhance the ability to capture temporal dependencies in multivariate time series data. The Autoencoder comprises an encoder and a decoder, each consisting of two layers. The outer layer contains 64 Bi-LSTM-WMC units, while the inner layer includes half as many units as the outer layer. The decoder reconstructs the input data, and the model is trained by minimizing the reconstruction error between the original input and its reconstruction. Upon completing the training, the reconstruction errors of the training data are calculated and used to establish the threshold for anomaly detection in subsequent steps. Figure 4 illustrates the general flow of our proposed model. Figure 5 illustrates the details of Bi-LSTM-WMCs cell in the model.

Step 3: Offline Detection

In this study, “normal” events are defined as patterns or behaviors in the data that reflect expected or typical operational conditions, while “anomalies” are deviations from these patterns, potentially indicating faults, errors, or unexpected behaviors. The definitions of “normal” and “anomalies” are dataset-specific. A detailed explanation of what constitutes “normal” and “anomalies” will be provided when each dataset is introduced.

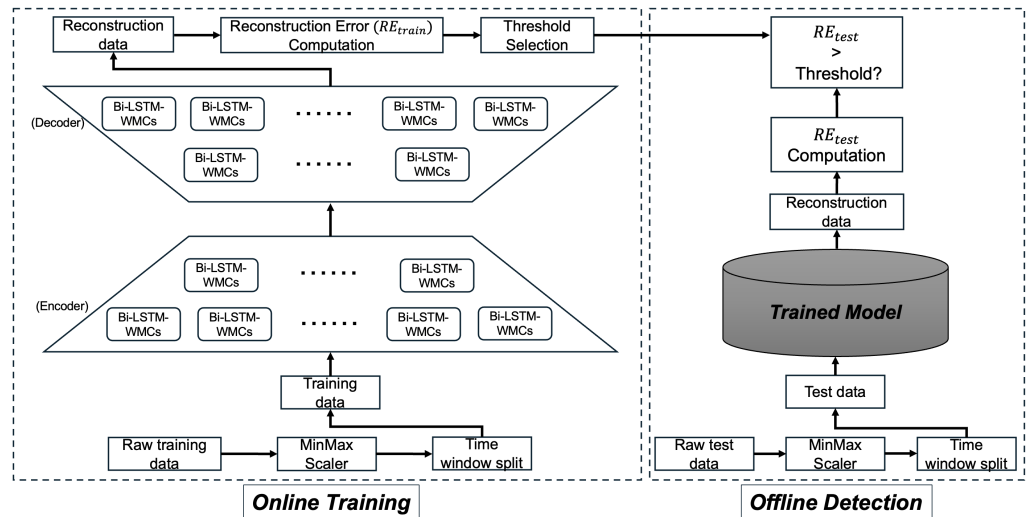


Figure 4. Overall process of the proposed approach.

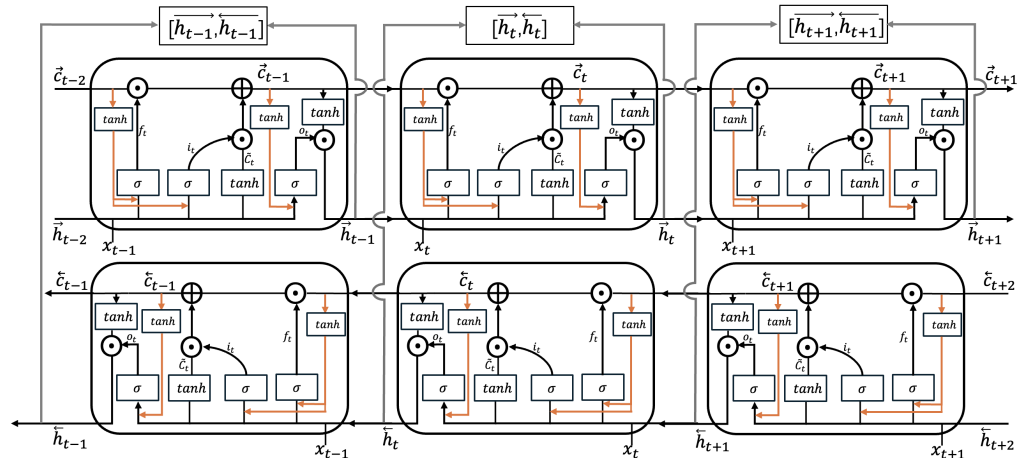


Figure 5. Detail of the Bi-LSTM-WMCs block.

An observation that deviates from the majority of the data might be referred to as an anomaly [29]. Prior to the anomaly detection phase, a threshold for distinguishing between normal and anomalous instances must first be established. Similar to traditional AE-based approaches, we use the absolute difference between the input data and the reconstructed data as the reconstruction error. However, it is important to note that since we are dealing with multivariate time series data, both the input and the reconstructed output of our model are three-dimensional arrays, where each time window is represented as a two-dimensional array. Each time point in the original dataset appears in multiple time windows. Therefore, to calculate the final reconstruction error for each time point, we sum all reconstruction errors for that time point and divide by the total number of time windows in which that time point appears. This average value is used as the final reconstruction error for the time point. Our model is trained on a training dataset that exclusively consists of normal data. It can provide us the reconstruction error values related to the normal data points only. After completing the model training and calculating all reconstruction errors, the threshold for anomaly detection is set as the $(1 - \alpha) \cdot 100$ percentile of the reconstruction errors from the entire training dataset. Here, α represents the confidence level commonly used in statistical methods, indicating the degree of error tolerance in our model. In the case study below, we will plot Performance Curves for α ranging from 0 to 0.1, with intervals of 0.01, to compare the model's performance at different α values. After determining the threshold, the test dataset, which includes both normal and anomaly data, is fed into the

trained model for anomaly detection. For each sample in the test dataset, the reconstruction error is calculated. If the reconstruction error exceeds the threshold, the sample is classified as anomalies.

4. Case Study

Our experiments were conducted on Google Colab, utilizing the default CPU environment with 12.7 GB of system RAM. The implementation was developed in Python 3.10, with key libraries including TensorFlow 2.15, Keras 2.15, Scikit-learn 1.2.2, and pandas 2.0.3, along with other relevant libraries.

To evaluate the performance of our proposed model, we applied three open-source datasets. We express our sincere gratitude to the authors who provided the datasets free of charge. The first one is the Skoltech Anomaly Benchmark (SKAB) dataset, which can be found on Kaggle [30]. The datasets contain a multivariate time series collected from the sensors installed on the testbed. Columns in each data file are as follows:

- **datetime**—Represents dates and times of the moment when the value is written to the database (YYYY-MM-DD hh:mm:ss);
- **Accelerometer1RMS**—Shows a vibration acceleration (amount of g units);
- **Accelerometer2RMS**—Shows a vibration acceleration (amount of g units);
- **Current**—Shows the amperage on the electric motor (ampere);
- **Pressure**—Represents the pressure in the loop after the water pump (bar);
- **Temperature**—Shows the temperature of the engine body (the degree Celsius);
- **Thermocouple**—Represents the temperature of the fluid in the circulation loop (the degree Celsius);
- **Voltage**—Shows the voltage on the electric motor (volt);
- **RateRMS**—Represents the circulation flow rate of the fluid inside the loop (liter per minute);
- **anomaly**—Shows if the point is anomalous (0 or 1);
- **changepoint**—Shows if the point is a changepoint for collective anomalies (0 or 1).

In the SKAB dataset, “normal” corresponds to system operation under expected conditions, where sensor readings remain within predefined ranges, whereas “anomalies” are manually induced faults, such as sensor malfunctions or component failures, intentionally introduced to cause deviations from expected values. The training data of the dataset contain normal data only. Also, since our study is only for anomaly detection and not for changepoint detection, the last column (changepoint) will be removed in the data preprocessing stage. The size and details of the dataset are summarized in Table 1. Figure 6 illustrates a subset of the training data and one of the test datasets from the SKAB dataset, with the red regions in the figure labeled as anomalies.

Table 1. Details of SKAB datasets.

Dataset	Data Shape	Data Amount	# of Anomalies	Anomaly Rate
Training dataset	(9405, 8)	9405	0	0
Test dataset 1	(1155, 8)	1155	410	0.3550
Test dataset 2	(1147, 8)	1147	402	0.3505
Test dataset 3	(1090, 8)	1090	347	0.3183

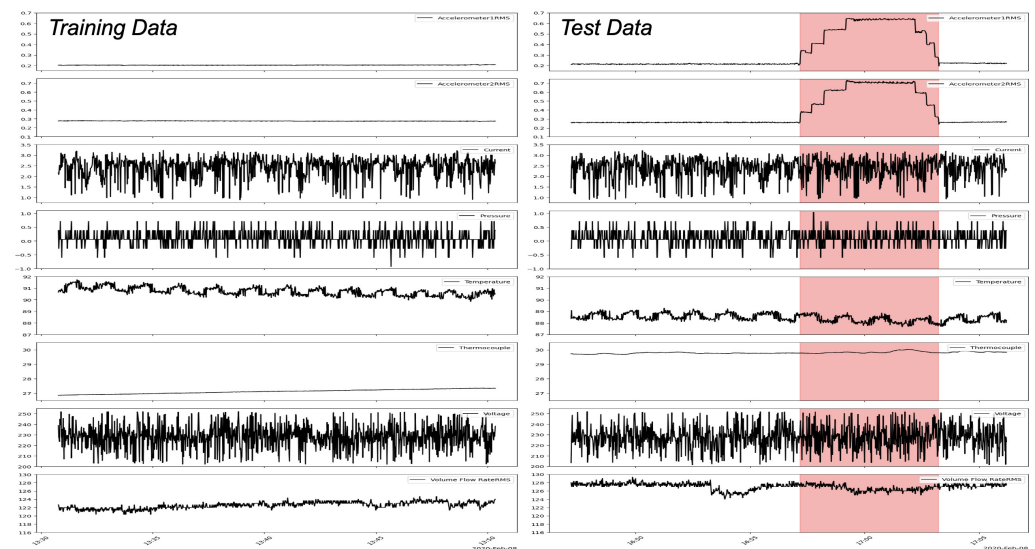


Figure 6. Visualization of a portion of the training data and one of the test datasets.

The second and third datasets are the Soil Moisture Active Passive (SMAP) and Mars Science Laboratory (MSL) datasets from NASA [31]. These datasets are also publicly available and contain multivariate time series collected from satellite and space exploration missions. They are widely used for benchmarking anomaly detection models due to their diverse set of anomaly types and realistic operational scenarios. For the SMAP dataset, “normal” represents the normal operation of the satellite’s soil moisture monitoring system, where telemetry readings align with expected operational conditions. “Anomalies,” on the other hand, are events derived from expert-labeled Incident Surprise Anomaly (ISA) reports, including system failures such as unexpected sensor readings or communication errors. Similarly, in the MSL dataset, “normal” denotes the proper functioning of the Mars Science Laboratory during its mission, as indicated by telemetry data that reflect standard operational behavior. “Anomalies” are identified through expert analysis of ISA reports and include irregularities such as sensor faults, communication disruptions, or mechanical malfunctions. Since the SMAP and MSL datasets contain multiple entities, each with corresponding training and testing data, we train and test each entity separately. Figure 7 shows the visualization of a subset of variables from the SMAP dataset (D-1). Figure 8 shows the visualization of a subset of variables from the MSL dataset (F-5). As in Figure 6, the red regions in the figure are labeled as anomalies.

We employ the commonly used F1-score as the benchmark for evaluating model performance. To assess the efficacy of our proposed model, we compared it with state-of-the-art models including Recurrent Neural Networks–Autoencoder (RNN-AE), LSTM-AE, LSTM-WMCs-AE, and Bi-LSTM-AE. The construction and parameters of all models are kept consistent with our proposed model. Additionally, we conducted extensive performance comparisons using different threshold values, with α ranging from 0 to 0.1 in increments of 0.01. To further assess the suitability of our proposed model for diverse requirements, we experimented with various time window sizes. This approach aims to determine the scenarios in which our model demonstrates significant advantages.



Figure 7. Visualization of a subset of variables from the SMAP dataset (D-1).

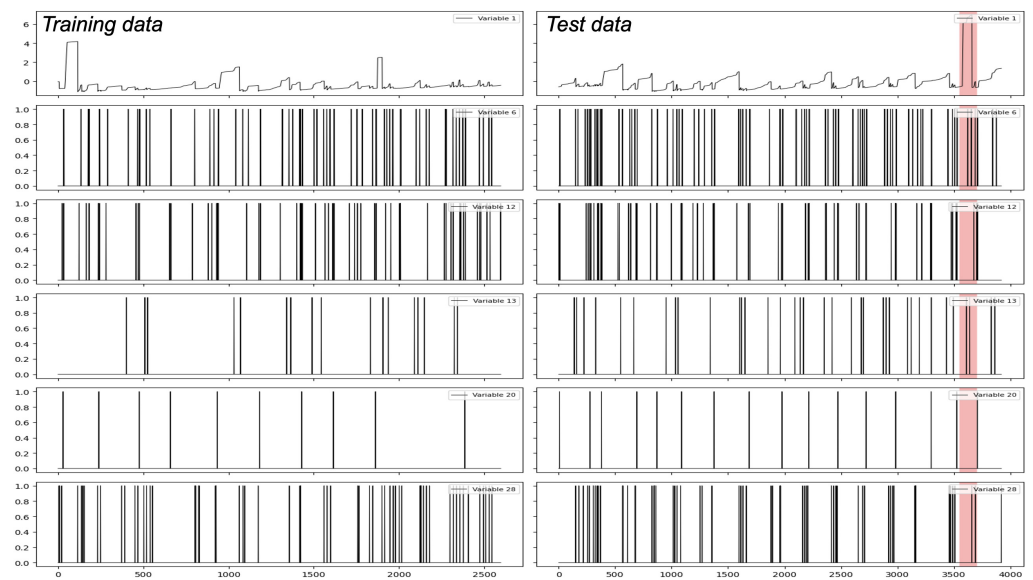


Figure 8. Visualization of a subset of variables from the MSL dataset (F-5).

For the SKAB dataset, we performed these comparisons as described above. However, for the other two datasets, due to space constraints, we selected the optimal time window size after multiple adjustments and compared the model's performance at α values of 0.01, 0.05, and 0.1. The calculation of F1-score is expressed using the following equations:

$$\text{Precision} = \frac{TP}{TP + FP} \quad (13)$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (14)$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \quad (15)$$

Here, TP (True Positive) refers to the count of accurately identified anomalies, TN (True Negative) refers to the count of correctly identified normal events, FP (False Positive) indicates the count of normal events that were inaccurately diagnosed as anomalies, while FN (False Negative) denotes the count of anomalies that were incorrectly identified as

normal events. Using Equations (13)–(15), we computed the performance of each model under various α values and time window sizes. Figure 9 illustrates the comparison of model performance across different time window sizes on the SKAB dataset. Table 2 presents the comparisons of model performance for four entities from the SMAP dataset. Table 3 shows the performance of the model on the MSL dataset. In this context, the varying α values determine the sensitivity thresholds for anomaly detection.

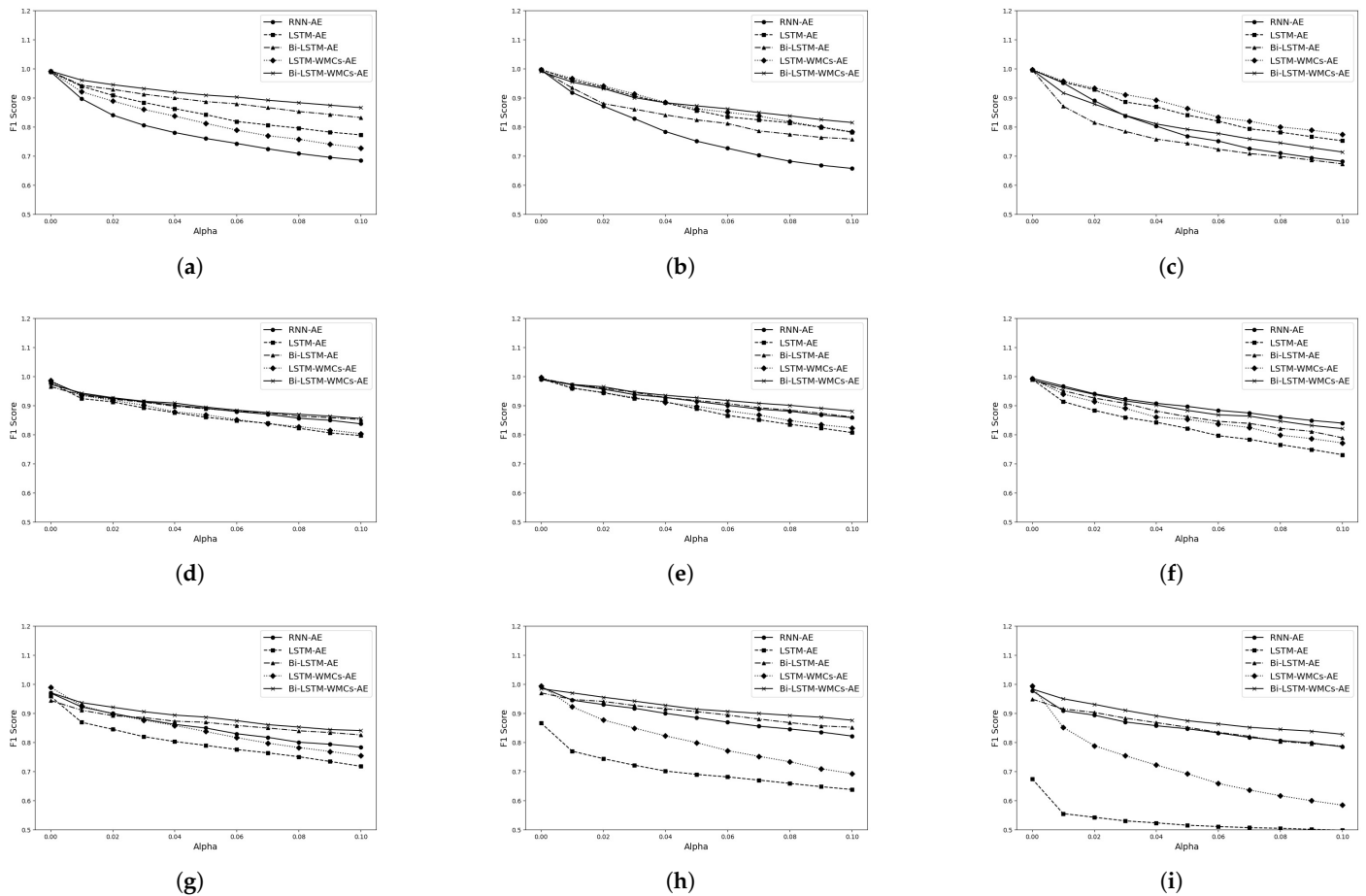


Figure 9. F1-score comparison for different window size (t) values and α . (a)–(c) represent window size (t) = 5 for 3 test datasets, (d)–(f) represent window size (t) = 25 and (g)–(i) represent window size (t) = 40.

To assess the feasibility of the proposed anomaly detection model in real-world scenarios, we conducted comparative experiments to evaluate the training time and detection time of various models across different datasets. Figure 10 illustrates the training time of each model for different datasets, while Figure 11 presents the time required for anomaly detection. Notably, our model utilizes a time window of size ($t \times m$) as input, meaning that an increase in the number of time points within a time window (t) or the number of variables in the data (m) inevitably increases computational complexity. This, in turn, necessitates longer training and fitting times. For example, in the SKAB dataset, when the time window size (t) is increased to 40, the training time for our proposed model is considerably longer than that of the baseline model. Although it increases complexity, it results in superior performance compared to the baseline model.

Table 2. Comparison of model performance for different subset of the SMAP dataset.

Dataset: D-1									
	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.1$		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
RNN-AE (Cho et al., 2014 [17])	0.3250	0.0080	0.0156	0.9081	0.9040	0.9060	0.8395	0.9086	0.8727
LSTM-AE (Jun et al., 2020 [18])	0.9569	0.6616	0.7822	0.9041	0.9052	0.9046	0.8307	0.9092	0.8682
Bi-LSTM-AE (Lee et al., 2022 [16])	0.9350	0.5655	0.7048	0.8958	0.9052	0.9005	0.8337	0.9119	0.8710
LSTM-WMCs-AE	0.9593	0.7748	0.8572	0.9005	0.9058	0.9032	0.8388	0.9116	0.8737
Bi-LSTM-WMCs-AE	0.9692	0.8969	0.9316	0.9025	0.9037	0.9031	0.8408	0.9107	0.8744
Dataset: D-2									
	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.1$		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
RNN-AE (Cho et al., 2014 [17])	0.9035	0.1221	0.2151	0.9422	0.9083	0.9249	0.8906	0.9130	0.9017
LSTM-AE (Jun et al., 2020 [18])	0.9586	0.5104	0.6662	0.9367	0.9092	0.9228	0.8921	0.9137	0.9028
Bi-LSTM-AE (Lee et al., 2022 [16])	0.9660	0.8295	0.8926	0.9197	0.9099	0.9148	0.8799	0.9137	0.8965
LSTM-WMCs-AE	0.9109	0.1332	0.2325	0.9358	0.9083	0.9218	0.8952	0.9130	0.9040
Bi-LSTM-WMCs-AE	0.9739	0.6799	0.8008	0.9422	0.9090	0.9253	0.8923	0.9130	0.9025
Dataset: D-4									
	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.1$		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
RNN-AE (Cho et al., 2014 [17])	0.9770	0.6542	0.7837	0.9092	0.8048	0.8538	0.8450	0.8966	0.8700
LSTM-AE (Jun et al., 2020 [18])	0.9447	0.6897	0.7973	0.9013	0.9276	0.9143	0.8488	0.9763	0.9081
Bi-LSTM-AE (Lee et al., 2022 [16])	0.9437	0.6807	0.7909	0.9034	0.8605	0.8814	0.8553	0.9430	0.8971
LSTM-WMCs-AE	0.9481	0.7928	0.8635	0.8988	0.9680	0.9321	0.8460	0.9840	0.9098
Bi-LSTM-WMCs-AE	0.9547	0.8765	0.9140	0.9058	0.9744	0.9389	0.8518	0.9821	0.9123
Dataset: P-2									
	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.1$		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
RNN-AE (Cho et al., 2014 [17])	0.8234	0.3842	0.5239	0.6707	0.4021	0.5028	0.5048	0.4250	0.4615
LSTM-AE (Jun et al., 2020 [18])	0.7544	0.4511	0.5646	0.6326	0.6305	0.6315	0.5628	0.6615	0.6082
Bi-LSTM-AE (Lee et al., 2022 [16])	0.7900	0.3744	0.5080	0.6387	0.5090	0.5665	0.5380	0.5889	0.5623
LSTM-WMCs-AE	0.8190	0.5685	0.6712	0.7085	0.6762	0.6920	0.5936	0.6958	0.6406
Bi-LSTM-WMCs-AE	0.8044	0.6207	0.7007	0.6450	0.6892	0.6664	0.5557	0.7080	0.6227

Table 3. Comparison of model performance for F-5 from the MSL dataset.

	$\alpha = 0.01$			$\alpha = 0.05$			$\alpha = 0.10$		
	Prec.	Rec.	F1	Prec.	Rec.	F1	Prec.	Rec.	F1
RNN-AE (Cho et al., 2014 [17])	0.6880	0.5695	0.6232	0.3514	0.6026	0.4439	0.2347	0.6093	0.3389
LSTM-AE (Jun et al., 2020 [18])	0.5316	0.5563	0.5437	0.2794	0.5828	0.3777	0.2083	0.5960	0.3087
Bi-LSTM-AE (Lee et al., 2022 [16])	0.5833	0.5563	0.5695	0.3560	0.5894	0.4439	0.2507	0.5960	0.3529
LSTM-WMCs-AE	0.4852	0.5430	0.5125	0.3347	0.5497	0.4160	0.2251	0.5695	0.3227
Bi-LSTM-WMCs-AE	0.8462	0.5828	0.6902	0.6794	0.5894	0.6312	0.3541	0.6026	0.4461

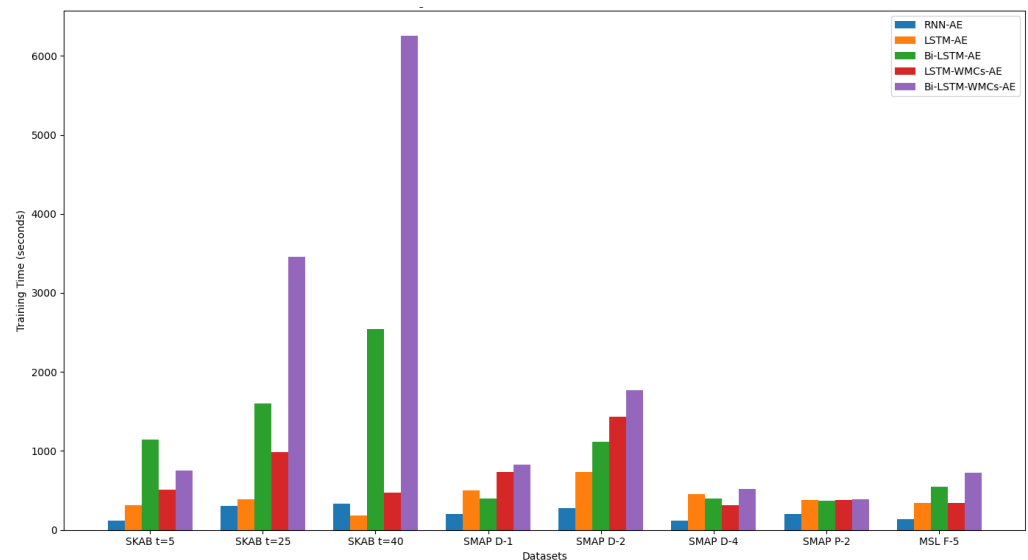


Figure 10. Comparison of training times across multiple models and datasets.

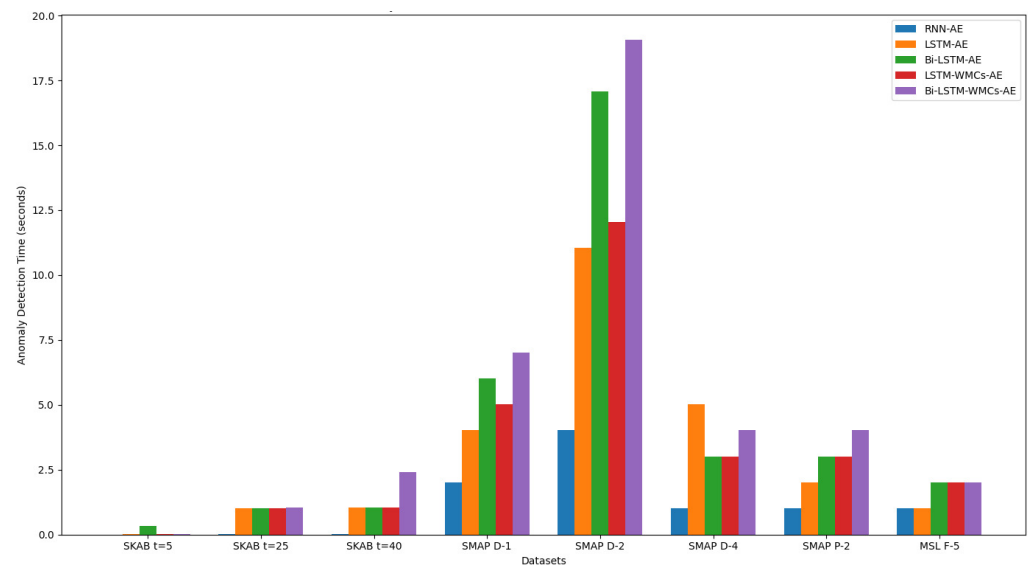


Figure 11. Comparison of anomaly detection times across multiple models and datasets.

Although the training time of our proposed model is longer than that of the baseline models—specifically, models utilizing this Working Memory Connections for LSTM take more time to fit compared to those based on conventional LSTM units—we argue that, with advancements in hardware, such computational resource consumption can be deemed acceptable, especially in applications where accuracy is critical, such as in nuclear power plant scenarios. Moreover, the difference in computational resource requirements is predominantly observed during the model training and fitting phase, with only negligible differences in the anomaly detection phase. Additionally, in real-time application scenarios, where each input consists of a single time window, the decision-making time for all compared models converges to nearly zero.

5. Conclusions

In this study, we developed a novel anomaly detection framework and demonstrated its application using several publicly available multivariate time series datasets. Our proposed framework combines the advantages of an AE and a new variant of LSTM, namely working memory connections. The AE part is responsible for learning the latent

features of the normal training data to acquire the ability to reconstruct normal data. When anomalous data are input into the trained model during the anomaly detection phase, the reconstruction error increases significantly. The Bi-LSTM-WMCs part is responsible for learning the temporal dependencies of the data. Through comparisons with a series of baseline models, we found that as the time window size (t) increases, the performance advantage of the model we proposed, namely Bi-LSTM-WMCs-AE, becomes increasingly evident. For smaller time windows ($t = 5$), the bidirectional architectures do not exhibit a significant advantage, likely due to the fact that with smaller time windows, unidirectional LSTM models can already effectively utilize the temporal information within the window. However, models using LSTM-WMCs (LSTM-WMCs-AE and Bi-LSTM-WMCs-AE) almost consistently outperform those without such connections. Therefore, we have reason to believe that our proposed model, incorporating bidirectional architecture and LSTM-WMCs, demonstrates greater robustness under various conditions.

The practical significance of our study lies in its ability to address the challenges of anomaly detection in multivariate time series data through a reconstruction-based approach. Central to our contribution is the integration of LSTM-WMCs, which, while not proposed by us, is applied for the first time in the anomaly detection domain in this study. This innovative LSTM variant enhances the model's ability to effectively process time series data by allowing memory cells to influence gate values, leading to more accurate reconstructions and improved anomaly detection reliability. This also demonstrates the feasibility of applying such a novel LSTM variant to other domains. Furthermore, our model significantly reduces the reliance on labeled anomaly data, addressing a critical limitation in real-world applications where such data are often scarce and expensive to obtain. The robust performance of our approach across datasets with varying characteristics, including continuous sensor data and mixed sensor-actuator data, further highlights its scalability and adaptability. These advancements validate the effectiveness of our approach and underscore its potential to provide a reliable and flexible solution for anomaly detection across diverse fields, such as industrial monitoring and space exploration.

Our study has some limitations that should be acknowledged. Firstly, while replacing the traditional LSTM units with the LSTM-WMCs has undoubtedly improved performance, this enhancement comes at the cost of increased computational complexity. The more intricate connection mechanism of the LSTM-WMCs demands greater computational resources, especially when combined with a bidirectional structure and integrated into the AE framework. This could pose challenges for deploying the model in resource-constrained environments or real-time applications, where computational efficiency is critical. Secondly, our study is primarily focused on detecting point anomalies, and the model's performance on other types of anomalies remains to be validated.

We have demonstrated the exceptional capability of the Working Memory Connections for LSTM in the domain of anomaly detection. However, LSTM has been widely employed in various real-world applications, such as logistics and inventory management, energy management, and intelligent production scheduling. An interesting direction for future research would be to investigate whether the Working Memory Connections for LSTM remains an effective tool in these application domains.

Author Contributions: Conceptualization, U.J.; methodology, X.D., J.W., Y.L. and U.J.; software, X.D.; validation, X.D., J.W., Y.L. and U.J.; formal analysis, X.D.; investigation, X.D., J.W., Y.L. and U.J.; resources, U.J.; data curation, X.D.; writing—original draft preparation, X.D. and J.W.; writing—review and editing, U.J.; visualization, X.D.; supervision, U.J.; project administration, U.J.; funding acquisition, U.J. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the Dongguk University Research Fund of 2025.

Data Availability Statement: The SKAB dataset used in this study can be accessed at the following link: <https://github.com/waico/SKAB> (accessed on 2 May 2024). The SMAP dataset used in this study can be accessed at the following link : <https://www.kaggle.com/datasets/patrickfleith/nasa-anomaly-detection-dataset-smap-msl> (accessed on 12 October 2024). The MSL dataset used in this study can be accessed at the following link: <https://www.kaggle.com/datasets/patrickfleith/nasa-anomaly-detection-dataset-smap-msl> (accessed on 12 October 2024).

Acknowledgments: The authors would like to thank the anonymous reviewers for their constructive comments.

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

Abbreviations

The following abbreviations are used in this manuscript:

LSTM	Long Short-Term Memory
Bi-LSTM	Bi-directional Long Short-Term Memory
AE	Autoencoder
LSTM-WMCs	LSTM with working memory connections
PCA	Principal Component Analysis
k-NN	k-Nearest Neighbors
LOF	Local Outlier Factor
KPCA	Kernel Principal Component Analysis
DBSCAN	Density-based spatial clustering of applications with noise
IoT	Internet of Things
RNNs	Recurrent Neural Networks
GRU	Gated Recurrent Unit
LSTM-AE	Long Short-Term Memory Autoencoder
GRU-AE	Gated Recurrent Unit Autoencoder
VAE	Variational Autoencoder
WMCs	Working Memory Connections

References

- Chandola, V.; Banerjee, A.; Kumar, V. Anomaly detection: A survey. *ACM Comput. Surv. (CSUR)* **2009**, *41*, 1–58. [\[CrossRef\]](#)
- Said Elsayed, M.; Le-Khac, N.A.; Dev, S.; Jurcut, A.D. Network anomaly detection using LSTM based autoencoder. In Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, Alicante, Spain, 16–20 November 2020, pp. 37–45.
- Habeeb, R.A.A.; Nasaruddin, F.; Gani, A.; Hashem, I.A.T.; Ahmed, E.; Imran, M. Real-time big data processing for anomaly detection: A survey. *Int. J. Inf. Manag.* **2019**, *45*, 289–307. [\[CrossRef\]](#)
- Shewhart, W.A. Economic quality control of manufactured product 1. *Bell Syst. Tech. J.* **1930**, *9*, 364–389. [\[CrossRef\]](#)
- Grubbs, F.E. Procedures for detecting outlying observations in samples. *Technometrics* **1969**, *11*, 1–21. [\[CrossRef\]](#)
- Surace, C.; Worden, K.; Tomlinson, G.; et al. A novelty detection approach to diagnose damage in a cracked beam. In Proceedings of the Proceedings-SPIE The International Society For Optical Engineering, Citeseer, San Diego, CA, USA, 30 July–1 August 1997; pp. 947–953.
- Liu, J.P.; Weng, C.S. Detection of outlying data in bioavailability/bioequivalence studies. *Stat. Med.* **1991**, *10*, 1375–1389. [\[CrossRef\]](#)
- Ye, N.; Chen, Q. An anomaly detection technique based on a chi-square statistic for detecting intrusions into information systems. *Qual. Reliab. Eng. Int.* **2001**, *17*, 105–112. [\[CrossRef\]](#)
- Samariya, D.; Thakkar, A. A comprehensive survey of anomaly detection algorithms. *Ann. Data Sci.* **2023**, *10*, 829–850. [\[CrossRef\]](#)
- Knorr, E.M.; Ng, R.T.; Tucakov, V. Distance-based outliers: Algorithms and applications. *Vldb J.* **2000**, *8*, 237–253. [\[CrossRef\]](#)
- Breunig, M.M.; Kriegel, H.P.; Ng, R.T.; Sander, J. LOF: Identifying density-based local outliers. In Proceedings of the 2000 ACM SIGMOD International Conference on Management of data, Dallas, TX, USA, 16–18 May 2000; pp. 93–104.
- Mika, S.; Schölkopf, B.; Smola, A.; Müller, K.R.; Scholz, M.; Rätsch, G. Kernel PCA and de-noising in feature spaces. *Adv. Neural Inf. Process. Syst.* **1998**, *11*, 536–542.

13. Sheridan, K.; Puranik, T.G.; Mangortey, E.; Pinon-Fischer, O.J.; Kirby, M.; Mavris, D.N. An application of dbSCAN clustering for flight anomaly detection during the approach phase. In Proceedings of the AIAA Scitech 2020 Forum, Orlando, FL, USA, 6–10 January 2020; p. 1851.
14. Zamanzadeh Darban, Z.; Webb, G.I.; Pan, S.; Aggarwal, C.; Salehi, M. Deep learning for time series anomaly detection: A survey. *ACM Comput. Surv.* **2024**, *57*, 1–42. [\[CrossRef\]](#)
15. An, J.; Cho, S. Variational autoencoder based anomaly detection using reconstruction probability. *Spec. Lect.* **2015**, *2*, 1–18.
16. Lee, S.; Jin, H.; Nengroo, S.H.; Doh, Y.; Lee, C.; Heo, T.; Har, D. Smart metering system capable of anomaly detection by bi-directional LSTM autoencoder. In Proceedings of the 2022 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 7–9 January 2022; pp. 1–6.
17. Cho, K. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078.
18. Jun, K.; Lee, D.W.; Lee, K.; Lee, S.; Kim, M.S. Feature extraction using an RNN autoencoder for skeleton-based abnormal gait recognition. *IEEE Access* **2020**, *8*, 19196–19207. [\[CrossRef\]](#)
19. Park, D.; Hoshi, Y.; Kemp, C.C. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. *IEEE Robot. Autom. Lett.* **2018**, *3*, 1544–1551. [\[CrossRef\]](#)
20. Raihan, A.S.; Ahmed, I. A Bi-LSTM Autoencoder Framework for Anomaly Detection-A Case Study of a Wind Power Dataset. In Proceedings of the 2023 IEEE 19th International Conference on Automation Science and Engineering (CASE), Auckland, New Zealand, 26–30 August 2023; pp. 1–6.
21. Gers, F.A.; Schmidhuber, J. Recurrent nets that time and count. In Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks, IJCNN 2000, Neural Computing: New Challenges and Perspectives for the New Millennium, Como, Italy, 27 July 2000; Volume 3; pp. 189–194.
22. Landi, F.; Baraldi, L.; Cornia, M.; Cucchiara, R. Working memory connections for LSTM. *Neural Netw.* **2021**, *144*, 334–341. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Jeong, Y.; Yang, E.; Ryu, J.H.; Park, I.; Kang, M. Anomalybert: Self-supervised transformer for time series anomaly detection using data degradation scheme. *arXiv* **2023**, arXiv:2305.04468.
24. Yang, Y.; Zhang, C.; Zhou, T.; Wen, Q.; Sun, L. Dcdetector: Dual attention contrastive representation learning for time series anomaly detection. In Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Long Beach, CA, USA, 6–10 August 2023; pp. 3033–3045.
25. Xu, H.; Wang, Y.; Jian, S.; Liao, Q.; Wang, Y.; Pang, G. Calibrated one-class classification for unsupervised time series anomaly detection. *IEEE Trans. Knowl. Data Eng.* **2024**, *36*, 5723–5736. [\[CrossRef\]](#)
26. Yong, B.X.; Brintrup, A. Do autoencoders need a bottleneck for anomaly detection? *IEEE Access* **2022**, *10*, 78455–78471. [\[CrossRef\]](#)
27. Hochreiter, S. Long Short-term Memory. In *Neural Computation*; MIT-Press: Cambridge, MA, USA, 1997.
28. Abduljabbar, R.L.; Dia, H.; Tsai, P.W. Unidirectional and bidirectional LSTM models for short-term traffic prediction. *J. Adv. Transp.* **2021**, *2021*, 5589075. [\[CrossRef\]](#)
29. Wei, Y.; Jang-Jaccard, J.; Xu, W.; Sabrina, F.; Camtepe, S.; Boulic, M. LSTM-autoencoder-based anomaly detection for indoor air quality time-series data. *IEEE Sensors J.* **2023**, *23*, 3787–3800. [\[CrossRef\]](#)
30. Katser, I.D.; Kozitsin, V.O. Skoltech Anomaly Benchmark (skab). Kaggle 2020. Available online: <https://github.com/waico/SKAB> (accessed on 1 March 2025).
31. Hundman, K.; Constantinou, V.; Laporte, C.; Colwell, I.; Soderstrom, T. Detecting spacecraft anomalies using lstms and nonparametric dynamic thresholding. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, London, UK, 19–23 August 2018; pp. 387–395.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.