Real-Time Monitoring Method for Chiller Unit's Cooling Water Inlet Flow Based on Time Series Classification

Fayang Li, Wen Yang, Aiping Pang and Qianchuan Zhao

Abstract—**The chiller serves as the core component of an industrial cooling system, with the cooling water inlet flow acting as a key indicator of the unit's operational status. As industrial production scales up and intelligent processes continue to evolve, traditional methods for real-time monitoring and anomaly detection of cooling water flow have proven inadequate due to their limitations in both accuracy and efficiency. We propose TCNAtten, a real-time monitoring approach based on time series classification (TSC), to address this challenge. This method classifies the collected cooling water flow data into different categories corresponding to actual operating conditions and utilizes the dilated convolutional structure of a Temporal Convolutional Network (TCN) to effectively capture multi-scale temporal features and dependencies across varying time scales. Furthermore, an attention mechanism is integrated to adaptively focus on the time steps that significantly influence the output, thereby enhancing the model's capability to capture long-term dependencies. Experimental results demonstrate that TCNAtten achieves superior accuracy and real-time responsiveness, enabling rapid and effective detection of changes in chiller flow rates. This research presents a novel intelligent monitoring and control solution in industrial cooling systems. It is important in advancing the intelligent management and optimization of industrial cooling systems.**

Index Terms—**time series classification, deep learning, real-time response, real-time monitoring.**

I. Introduction

With the acceleration of industrialization and the continuous expansion of production scales, real-time monitoring and data analysis of industrial equipment have become increasingly vital in modern industrial production. Equipment's health, operational efficiency, and safety are directly linked to production stability and economic performance. By continuously monitoring key operational parameters and analyzing real-time data, potential faults or anomalies can be detected early, preventing equipment downtime and damage and enhancing overall system efficiency. In this context, the chiller is a core component of industrial cooling systems, making its operational monitoring particularly critical. By circulating cooling water, chillers manage heat loads and ensure the stable operation of equipment and production processes. They are extensively employed across various sectors—manufacturing, chemical processing, electronics, food production, and pharmaceuticals—all requiring precise temperature control and high equipment reliability. For example, in manufacturing, chillers provide cooling for injection moulding machines and moulds; in the chemical industry, they regulate the temperature of reactors; in electronics, they cool laser devices; and in food and pharmaceutical industries, they maintain controlled production environments. The reliable performance of chiller units is essential for ensuring uninterrupted production, with the inlet flow of cooling water serving as a key indicator of operational status. Real-time monitoring of flow rate fluctuations enables the early identification of anomalies, allowing timely preventive interventions that improve efficiency and operational stability. However, traditional flow monitoring approaches have relied on manual inspections or experience-based threshold settings. While these methods offer basic monitoring capabilities, they often fail to accommodate the complexity and variability of modern industrial environments. Delayed response times, false alarms, and missed detections undermine monitoring accuracy and responsiveness. In contemporary industrial settings, the rapid advancement of Internet of Things (IoT) technologies has accelerated the transition toward intelligent factories [1]. The proliferation of sensors has resulted in exponential data growth, expansive data streams, and increasingly stringent real-time performance requirements. Constrained by limited data-handling capabilities, traditional monitoring methods frequently fall short of meeting the demands for efficient and accurate anomaly detection in complex industrial environments. Consequently, there is a pressing need for advanced technologies to overcome these limitations and enhance the precision and responsiveness of industrial monitoring systems.

In recent years, TSC methods based on machine learning and deep learning have increasingly emerged as effective tools for system monitoring. TSC is a technique used to analyze time series data, aiming to assign data samples from different time intervals to predefined categories. By training models to identify patterns and temporal trends, TSC methods enable the recognition of recurring changes within the data, thereby facilitating state prediction and anomaly detection. TSC has already been successfully applied across various domains, including structural health monitoring [2], marine gas emission tracking [3], and pavement damage

Manuscript received February 19, 2025; revised May 3, 2025.

This work was supported by the National Natural Science Foundation of China (Grant No.62192751 and 61425027), the Department of Education of Guizhou Province, QianJiaoJi, China [2022]043 and the Guizhou Provincial Postgraduate Research Fund [YJSKYJJ[2021]010].

Fayang Li is a Postgraduate of the School of Electrical Engineering, Guizhou University, Guiyang, 550025, China. (e-mail: fayangli@126.com).

Wen Yang is a Senior Engineer of the Key Laboratory of Space Launching Site Reliability Technology, Xichang Satellite Launch Center, Xichang, 615000, China. (Corresponding author to provide phone: +86-185-1088-2704; e-mail: whutyw@126.com).

Aiping Pang is a Professor of the School of Electrical Engineering, Guizhou University, Guiyang, 550025, China. (Corresponding author to provide phone: +86-156-4506-2396; e-mail: appang@gzu.edu.cn).

Qianchuan Zhao is a Professor of the Department of Automation, Tsinghua University, Beijing, 100084, China. (e-mail: zhaoqc@tsinghua.edu.cn).

detection [4]. By leveraging large volumes of historical data, these methods automatically learn equipment operational patterns and distinguish between normal and abnormal states, thereby improving monitoring systems' accuracy and reliability. In particular, deep learning-based models—such as Recurrent Neural Networks (RNN) [5] and Convolutional Neural Networks (CNN) [6]—have demonstrated notable advantages in handling time series data, especially when dealing with complex nonlinear dynamics and high-dimensional datasets. Among them, the TCN [7] has emerged as a powerful architecture specifically designed to address the unique characteristics of time series data by enhancing the structure of conventional CNN. Its structural design is illustrated in Fig. 1. TCN utilizes causal convolution, which ensures that future information does not influence past predictions, preserving the integrity of time-dependent forecasting. Moreover, dilated convolution expands the receptive field, enabling the model to capture long-range dependencies effectively. Incorporating residual connections mitigates the vanishing gradient problem commonly encountered in deep networks and significantly improves training efficiency through parallel computation. Compared to RNN-based models, TCN offers superior computational efficiency and more stable gradient propagation while maintaining the capacity to model temporal dynamics. These advantages make TCN well-suited for real-time processing of high-speed data streams in industrial applications.

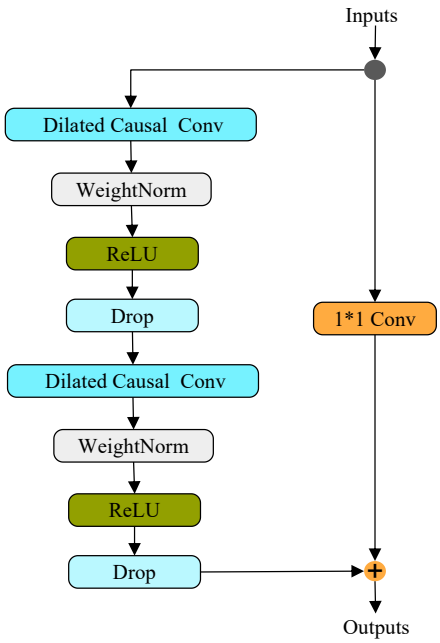

Fig. 1. Structures of TCN

Despite significant advancements in anomaly detection methods based on TSC, several challenges remain in practical industrial applications. First, traditional classification approaches rely on offline analysis and predefined feature extraction. While these methods perform adequately in static or slowly evolving environments, their computational efficiency and responsiveness often become major bottlenecks when dealing with massive real-time data streams and meeting stringent decision-making requirements. Furthermore, traditional models struggle to adapt to emerging anomaly patterns in dynamically changing operational contexts, resulting in limited flexibility and adaptability in real-world applications. Second, although deep learning models have demonstrated exceptional capabilities in handling complex data structures, their high computational complexity and prolonged training times significantly hinder their deployment in industrial monitoring systems where real-time performance is paramount. When faced with large-scale data, these models' substantial computational resource demands further constrain their applicability in resource-limited industrial settings. Consequently, there is growing interest in developing efficient models that achieve high classification accuracy while reducing computational overhead and enhancing real-time responsiveness. Such models must be capable of processing large-scale real-time data streams efficiently and precisely while demonstrating high sensitivity to anomalous patterns to enable early fault detection and rapid response. This would allow for accurate fault warnings and significantly enhance system stability and operational efficiency. Designing an efficient real-time monitoring model is essential for improving the performance of industrial monitoring systems and represents a crucial step toward intelligent system evolution. Moreover, it is a prerequisite for achieving safer, more efficient, smarter industrial production environments.

To address these challenges, we propose TCNAtten, a deep learning-based TSC method for real-time monitoring of chiller cooling water inlet flow. The proposed approach begins by collecting historical inlet flow data, followed by essential preprocessing steps such as denoising and normalization. These procedures are designed to eliminate noise interference and standardize data scales, ensuring high quality and usability. Subsequently, the data are categorized according to actual operational conditions to establish a reliable foundation for downstream modelling and analysis. Temporal features within the flow data are extracted using a TCN, which, through its deep convolutional architecture and dilated convolution mechanism, effectively models long-range dependencies and identifies intricate patterns in time series data. Furthermore, an attention mechanism [8] is integrated to enhance the model's ability to automatically focus on the most informative aspects of the input, thereby improving sensitivity to abnormal fluctuations in high-dimensional data. This integration strengthens the model's capacity to detect anomalous states accurately. By combining TCN with an attention mechanism, the proposed TCNAtten model efficiently captures temporal dynamics and dynamically adjusts its focus based on the relative importance of different data points. This enables precise identification of abnormal fluctuations, facilitating real-time monitoring and fault detection in chiller systems. Experimental results demonstrate that TCNAtten outperforms the state-of-the-art TSC method HIVE-COTE 2.0 (HC2) [9] and the high-speed classification algorithm MiniRocket [10] in terms of classification accuracy, real-time performance, and robustness. The model effectively addresses the nonlinear characteristics and temporal dependencies inherent in flow data, enabling real-time anomaly detection and providing valuable support for fault prediction and maintenance decision-making. These capabilities significantly enhance system efficiency and operational stability, underscoring the model's strong potential for practical deployment in industrial monitoring applications.

II. Related Work

In the real-time monitoring of chiller units, traditional methods—such as threshold-based settings, rule-based reasoning, and expert-driven approaches—have played an important role in basic fault detection and system monitoring. However, as system complexity increases and environmental conditions become more dynamic, these conventional techniques increasingly reveal their limitations. Specifically, they often fail to effectively handle nonlinear faults, fluctuating operating conditions, and variable loads. Furthermore, in situations involving multiple simultaneous faults, dynamically changing equipment states, and intricate interactions among environmental factors, fixed-threshold and rule-based monitoring approaches struggle to provide timely responses and lack the adaptability and robustness required for reliable system oversight. Consequently, there is a pressing need for intelligent, real-time monitoring technologies that can adapt to complex and evolving operational conditions to ensure the efficient performance and health management of chiller systems.

The core of modern real-time monitoring technology lies in the dynamic acquisition of various chiller sensor data—such as temperature, pressure, flow rate, and power consumption—followed by real-time analysis and feedback to detect abnormalities or operational failures promptly. With technological advancements, current monitoring systems extensively leverage various advanced tools. For example, the cooling efficiency of a chiller unit can be estimated and monitored using soft sensors [11]. However, this method typically requires evaluating multiple algorithms to identify the optimal solution, increasing implementation complexity. In addition, techniques based on Principal Component Analysis (PCA) support dynamic monitoring and trend visualization of chiller performance [12]. Nonetheless, due to PCA's high computational cost, its real-time performance significantly degrades when applied to large feature sets or high-volume datasets, making it difficult to satisfy the requirements of high-frequency data streams. While these modern monitoring methods have improved system intelligence and addressed several shortcomings of traditional approaches, achieving an optimal balance among computational complexity, real-time responsiveness, and accuracy in large-scale data scenarios remains a major challenge. Against this backdrop, TSC techniques are emerging as a promising solution. By learning temporal features from historical data, TSC can construct a mapping between equipment states and time-series patterns, enabling real-time fault mode identification. Compared to traditional methods, TSC automatically extracts meaningful features from raw data and adapts more effectively to complex and evolving operating conditions. By leveraging extensive historical datasets, TSC can more accurately identify various fault modes and detect potential risks in advance, supporting early warning mechanisms. Moreover, TSC can capture both long-term dependencies and localized patterns in time-series data, enhancing the adaptability and reliability of monitoring systems operating in dynamic and complex industrial environments.

However, despite the considerable advantages of TSC methods in monitoring systems, they still face several challenges—particularly in real-time processing, computational complexity, and storage efficiency. Real-time monitoring systems must handle massive volumes of time-series data, especially when equipment operates continuously, or multiple devices are monitored concurrently. Efficiently storing and processing such large-scale data while ensuring rapid training and inference of classification models has become a critical concern. Traditional TSC approaches, including those based on Shapelets, dictionary learning, and distance metrics, often suffer from excessive computational demands. For instance, Shapelet-based methods [13], [14], [15], [16] classify time-series data by identifying key discriminative subsequences (Shapelets), often achieving high classification accuracy. However, their computational complexity grows exponentially with the number of candidate Shapelets, severely degrading real-time performance. Similarly, distance metric-based methods [17], [18], [19], which rely on similarity measures (e.g., Euclidean distance, Dynamic Time Warping (DTW)) for classification or clustering, are conceptually straightforward and theoretically robust. Nevertheless, they incur substantial computational overhead when applied to high-frequency data streams, limiting their real-time applicability. Dictionary-based methods [20], [21], [22] represent time-series data using a learned dictionary of feature patterns. While effective in certain contexts, these methods require frequent dictionary construction and updates in streaming environments, consuming significant computational resources and negatively impacting system performance. Feature-based methods [23], [24], which classify time-series data by extracting statistical features (e.g., mean, variance, skewness, kurtosis) or frequency-domain features (e.g., Fourier transform, Wavelet transform), offer advantages in specific scenarios. However, they still struggle with high data dimensionality, strict real-time constraints, and complex feature selection processes.

In recent years, deep learning-based TSC methods—such as CNN and Long Short-Term Memory (LSTM) networks—have attracted increasing attention due to their ability to learn complex patterns from time-series data. However, their practical deployment remains constrained by high computational costs and prolonged training times. For example, LSTM-based models [25] are effective at capturing long-term temporal dependencies, but they demand substantial computational resources, particularly when processing long-sequence data. Although CNN-based approaches [26] excel at extracting local features, real-time processing of high-dimensional time-series data continues to pose a significant challenge. Transformer-based methods [27], [28], which leverage self-attention mechanisms, are capable of modelling long-range dependencies while mitigating gradient vanishing and exploding problems commonly associated with traditional RNCNNNs and LSTMs. Nevertheless, they still suffer from high computational overhead, limited real-time performance, and substantial memory usage—especially in large-scale data stream scenarios with strict latency requirements.

Researchers have also attempted to adopt ensemble methods to improve classification performance and model generalization. Such as HC2, combine multiple classifiers—including Temporal Dictionary Ensemble (TDE) [29], Diverse Representation Canonical Interval Forest (DrCIF) [9], Random Convolutional Kernel Transform (ROCKET) [30], and Shapelet Transform Classifier (STC) [31]—to en-

hance classification performance through weighted voting. However, training multiple sub-models incurs significant computational and storage overhead, adversely impacting real-time responsiveness. InceptionTime (InceptionT) [32] attempts to balance computational efficiency and classification accuracy by integrating five deep learning-based classifiers, thus improving TSC performance in real-time contexts. Meanwhile, MiniROCKET has emerged as one of the fastest classifiers, delivering extremely low-latency predictions. In contrast, although HC2 remains one of the most accurate classifiers, it suffers from high computational complexity, long training times for lengthy sequences, and slower inference speed. Several other state-of-the-art classifiers have also demonstrated strong performance in TSC. FreshPRINCE [33], a feature-based method, effectively captures discriminative characteristics from time-series data. MultiROCKET [34], a convolutional kernel-based approach, efficiently extracts local features using an optimized convolutional kernel structure. Hydra [35], a hybrid dictionary–ROCKET architecture, allows flexible replacement of ROCKET with MiniROCKET or MultiROCKET to adjust for speed and accuracy. The Random Dilated Shapelet Transform (RDST) [36], a shapelet-based classifier, identifies key subsequences to enhance classification precision. WEASEL 2.0 [37], a dictionary-based method, decomposes time series into subsequences or symbolic vocabularies and uses their frequency distributions for classification.

In summary, although TSC methods provide a more intelligent and efficient solution for real-time monitoring, their widespread adoption in practical applications remains constrained by challenges related to real-time performance, computational complexity, and storage demands. Future research should focus on improving algorithmic efficiency, reducing computational overhead, and enhancing model adaptability and robustness to meet the growing requirements for real-time processing, high accuracy, and scalability in industrial environments—while maintaining classification precision. By integrating multidisciplinary approaches such as edge computing, distributed computing, and lightweight model design, TSC technologies can be further advanced to enable broader deployment in industrial monitoring systems. These advancements would offer strong technical support for the intelligent operation and maintenance of chillers and other industrial equipment, ultimately enhancing overall system efficiency and reliability.

III. TCNAtten Framework

To enable real-time monitoring of the cooling water inlet flow rate in chiller systems, this paper proposes and develops a time series classification model based on deep learning, the TCN-Attention (TCNAtten) model. Fig. 2(a) illustrates the model's overall architecture. This model integrates the local feature extraction capabilities of TCN with the global dependency modelling power of a multi-head attention mechanism, making it highly effective for analyzing complex time series data. In the TCN component, dilated convolutions enlarge the receptive field, allowing the model to capture long-range temporal dependencies. The model gradually expands its perceptive scope by introducing varied dilation rates, enabling it to detect dynamic changes over extended time spans without incurring significant computational overhead. This design empowers the model to process time series from a multi-scale perspective, effectively extracting critical features while maintaining computational efficiency. To address the response delay typically associated with the directional constraint of causal convolutions, the model intentionally omits causal convolutions. This decision preserves the continuity and integrity of the time series data and facilitates bidirectional feature interaction across network layers, enhancing the model's sensitivity to abrupt or anomalous signals. Finally, a multi-head attention mechanism is incorporated to strengthen the model's ability to capture global dependencies. The model can identify complex temporal relationships within different subspaces by conducting parallel computations across multiple attention heads. This mechanism also dynamically assigns attention weights to different time steps, allowing the model to focus on the most informative moments in the sequence and ultimately improving prediction accuracy.

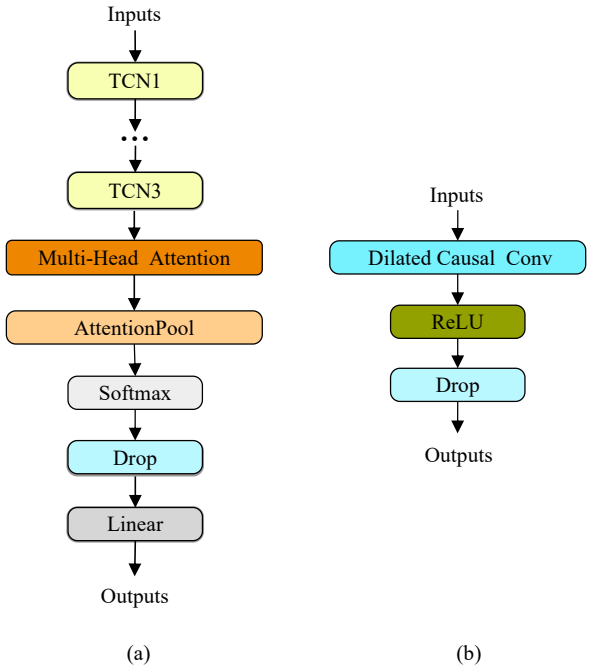

Fig. 2. Structures of TCNAtten (a) and TCN (b)

In TCNAtten, the core component is the Temporal Convolutional Network (TCN) layer, as illustrated in Fig. 2(b). The TCN layer efficiently models multi-scale temporal dependencies by progressively increasing the dilation rate across layers. Owing to the intrinsic structure of dilated convolutions, the model supports parallel computation, enhancing computational efficiency. Moreover, each layer incorporates the ReLU activation function and dropout operation, strengthening the model's nonlinear representation capacity while reducing the risk of overfitting. The mathematical formulation of dilated convolution is defined as follows:

$$y_t = \sum_{i=0}^{k-1} w_i \cdot x_{t-i \cdot d} \tag{1}$$

where y_t is the output at time step t, k is the convolution kernel size,w_i is the weight of the i-th convolutional kernel,

$x_{t-i\cdot d}$ is the input of the $t-i\cdot d$-th time step, and d is the expansion coefficient.

To capture global dependencies between time steps in a time series, TCNAtten incorporates a multi-head attention mechanism. This mechanism enhances the modelling of global dependencies by computing multiple attention heads in parallel, allowing it to extract information from different subspaces within the sequence. Each attention head is computed as follows:

$$Attention(Q,K,V)=softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{2}$$

where Q, K, and V represent the query, key, and value matrices, respectively, and d_k is the dimension of the key used to scale the dot product. By computing multiple attention heads in parallel, the model can simultaneously capture different types of time-series relationships. Finally, the outputs of multiple attention heads are concatenated and linearly transformed (via the matrix W_o) to obtain the final attention output:

$$MultiHead\left(Q,K,V\right)=Concat\left(head_1,...,head_h\right)W^{\circ} \tag{3}$$

where $head_i = Attention(QW_i^Q, KW_i^K, VW_i^V)$,$QW_i^Q$ KW_i^K and VW_i^V are the linear transformation matrix for each attention head.

After passing through the TCN and multi-attention layers, the TCNAtten model performs element-wise multiplication and weighted summation of the TCN and multi-attention outputs. This process enables the model to focus on the most important information in the sequence while assigning different weights to features at various time steps, enhancing its ability to capture time-series patterns effectively. The calculation formula is as follows:

$$W=O_{TCN}\odot O_{Atten} \tag{4}$$

where O_{TCN} is the TCN output, O_{Atten} is the multi-attention output and $\odot$ denotes the element-wise multiplication of the corresponding elements in the matrices.

To further enhance feature representation, TCNAtten incorporates an attention-pooling layer. Using a fully connected network, this layer computes attention weights for each time step. It generates a fixed-length feature vector by weighting and summing these weights, thereby aggregating the key information in the time series. Its advantage lies in the ability to dynamically select the most crucial time-step features for the task, capture long-term dependencies, reduce information loss, and enhance the model's interpretability and expressiveness. Compared to traditional pooling methods, the attention-pooling layer can extract important information from time-series data more precisely, thereby improving the model's performance in time-series tasks. The formula for the attention-pooling layer is:

$$a_i=softmax\left(W_2ReLU\left(W_1h_i\right)\right) \tag{5}$$

where W_1 and W_2 are the linear transformation matrices, and h_i represents the features at time step t in the sequence. Subsequently, the features of all time steps are weighted and summed. This process combines the features of each time step in the time series according to their corresponding attention weights, and the resulting features can be expressed as:

$$z=\sum_{i=1}^{T}a_ih_i \tag{6}$$

where a_i is the attentional weight (the weight obtained after attentional pooling) for time step i.

A Dropout operation is applied before the fully connected layer to prevent overfitting. Dropout randomly drops certain features, thereby enhancing the model's generalization ability. Finally, the classification results are transformed into a probability distribution over the target classes using the Softmax function:

$$\hat{y}=soft\max\left(W_f\cdot z\right) \tag{7}$$

where W_f is the weight matrix of the classification layer and $\hat{y}$ is the predicted categorical probability distribution.

In summary, TCNAtten efficiently captures key patterns in time series by combining a temporal TCN with a multi-head attention mechanism. Its innovations are primarily reflected in the following aspects: First, the inflated convolutional structure of the TCN allows the model to effectively capture local features and multi-scale dependencies within the time series, thereby enhancing its ability to learn complex temporal patterns. Second, incorporating the multi-head attention mechanism further improves the model's ability to model long-range dependencies, enabling it to better capture global patterns in the time series. The model adaptively assigns different weights to various time steps through the attention mechanism, automatically focusing on the most important information, which significantly boosts classification performance. Additionally, the element-wise multiplication of the TCN's output with the attention output, similar to the Shapelet extraction process, enables the model to focus on the most relevant information in the sequence. This combined approach leverages the attention mechanism to dynamically assign weights to the features extracted by the TCN, thereby facilitating the more accurate capture of significant patterns in the time series. Finally, introducing the attention pooling layer enables the model to adaptively aggregate key information from the time series, further enhancing its feature selection capability and improving classification accuracy.

IV. Experiment

The experiment focuses on the historical data of the cooling water inlet flow sensor of Chiller Unit 1 in a data centre. The system structure diagram in Fig. 3 illustrates the refrigeration system, which consists of seven chillers, each with a cooling capacity of 4,550 kW (1,300 tons). Due to space limitations, the diagram only depicts the connection relationships of three chillers, while the remaining four follow the same configuration as the first three. The system operates in a "6+1" mode, with six chillers and one on standby. The training dataset was selected from March 31, 2016, to June 30, 2016, while the test dataset was selected from August 31, 2016, to October 31, 2016. All data were recorded every minute, covering the operating status of the chiller under different operating conditions.

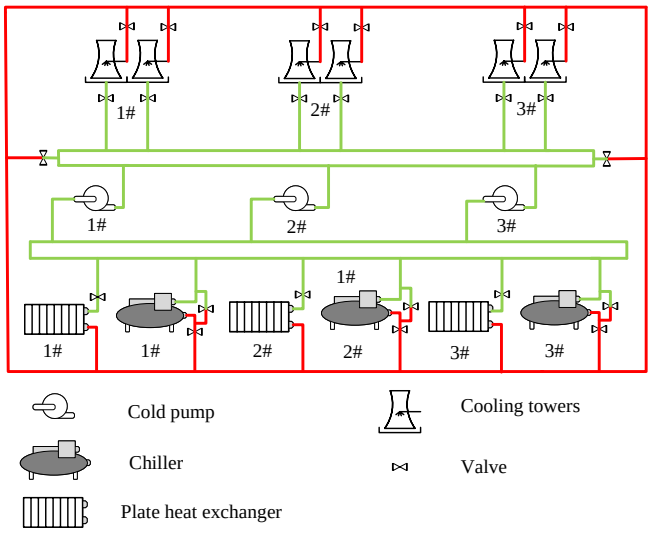

Fig. 3. Schematic Diagram of Chiller System

A. Data Preprocessing

Since sensor data is often affected by environmental changes, equipment malfunctions, and other factors, comprehensive preprocessing is crucial to ensure the quality and reliability of the data before it is used for training. The data preprocessing step includes handling missing values and identifying and correcting outliers to prevent these issues from negatively impacting the model's performance.

Missing values are common in sensor data and can be caused by sensor malfunctions or temporary failures of the acquisition device. We employed the following two strategies to address missing values:

1)If only one missing value is present at a given time, it is filled with the mean value of the data before and after that time. This method smooths the data and avoids introducing extreme values that could affect model training.

2)If multiple consecutive missing values are found over a period, they are padded with 0. Since the chiller generates certain data (e.g., flow data) even when turned off, padding with zero helps identify sensor failures or abnormal states.

In a device as complex as a chiller, anomalous data can result from faulty sensors, external interference, or data transmission issues. We developed different handling strategies based on various operating states:

1)In the shutdown state of the chiller, the unit should generally operate at a value of around 2. Therefore, any outliers are likely due to sensor failures or data acquisition errors. We verified the data by comparing each value. If a data point exceeded the set thresholds (1 and 4, respectively, based on historical operation patterns), it was considered an outlier and was corrected to the threshold.

2)Under normal operating conditions, the chiller typically maintains a relatively stable state, but occasional unreasonable fluctuations may occur. We used a sliding window (with a window length of 60 and a step size of 60) to move along the time series and calculate the mean of 50 data points, excluding the five highest and five lowest values, which were then used as representative values for the window. If a data point exceeded the mean by a factor of 1.08 or fell below 0.92 (thresholds set based on historical experience), it was treated as an outlier. The outlier was replaced with the largest or smallest value in the window, excluding the outlier.

Additionally, we applied data truncation measures for extreme values in the data. For outliers greater than 750, we truncated them to 750 to prevent these extreme values from affecting the training process. (The threshold of 750 was determined based on the data's historical running pattern, with few values exceeding this limit.) The steps of data preprocessing are illustrated in Fig. 4. The visualizations of the selected partial datasets before and after preprocessing are shown in Fig. 5 and Fig. 6, respectively.

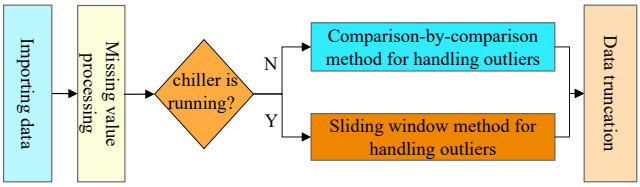

Fig. 4. After the data is exported from the data center, different methods are applied to handle missing values and anomalies at various stages of operation. Finally, data exceeding the feasible range is truncated

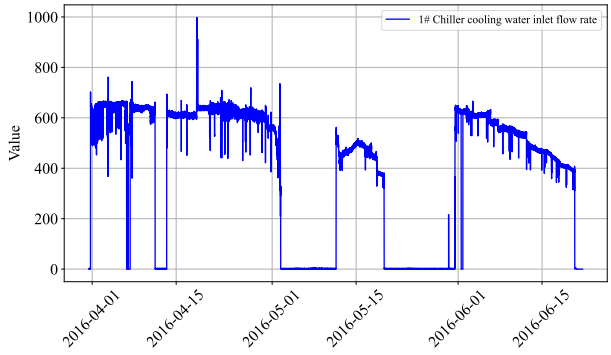

Fig. 5. Visualization of data before preprocessing

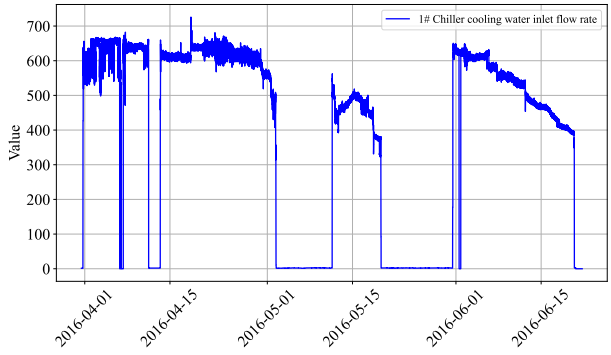

Fig. 6. Visualization of the data after preprocessing

B. Category Delineation

To perform effective TSC, we classify the chiller data according to different operating states and set appropriate time series lengths and window steps to ensure diversity and enable real-time training. Since the chiller takes approximately one hour to reach a steady state, we set the

length of each time series to 60 steps, corresponding to 1 hour of data. This ensures that each time series captures the complete dynamics of the unit. Based on the chiller's actual operating conditions, we categorize the data into different states: normal operation, warning status (e.g., yellow alarm), and shutdown status. Table I presents the specific categorization and the number of sequences in each category. Due to the extremely small number of red warning cases in the entire dataset (only one data point), this category is excluded from training and testing, as the imbalance in samples could negatively affect the model's learning in TSC tasks. Additionally, since no yellow alarm data was available for the test set period (August 31, 2016, to October 31, 2016), we substituted it with data from September 23, 2015, at 17:08 to September 30, 2015, at 10:28. This adjustment ensures that the model can be effectively evaluated for detecting the chiller's yellow alarm state. In the category segmentation process, the chiller is on/off state; due to the limited data availability in this state and its high real-time requirements, we apply a sliding window with a length of 60 and a step size of 1 to extract the time series. This ensures that each on/off event is accurately captured. For other categories, such as normal operation or alarm status, we use a sliding window with a step size of 60 to extract the time series. This approach helps to increase the diversity and coverage of the data by leveraging the abundant operational data available during the corresponding period.

TABLE I
INTRODUCTION TO THE CLASSIFICATION OF CATEGORIES AND THE NUMBER OF DIVISIONS

Status Description	Label	Training number	Test number
hiller is not running	0	271	199
Chiller is on	1	236	118
Chiller is off	2	236	118
Normal operation	3	206	690
Warning	4	202	117
Yellow warning	5	191	165

C. Experimental Comparison

The detailed parameter configuration of TCNAtten is presented in Table II. The model employs eight attention heads and is trained using the Adam optimizer with a learning rate 0.001. The loss function used is nn.CrossEntropyLoss(), and a dropout rate of 0.4, is applied to enhance generalization, with an additional dropout rate of 0.3 specifically applied within the TCN layers. The parameter settings for HC2 are listed in Table III. For the MiniRocket and MultiRocket models, 10,000 convolutional kernels were utilized, while all other models retained their default configurations. Experiments were conducted on a machine equipped with a 12th Gen Intel® Core™ i7-12700H 2.30 GHz processor and 16GB of RAM. Each model was independently executed 30 times. The experimental outcomes are illustrated in Fig. 7. The results show that, among the 1,047 data samples, Hydra achieved the fastest execution time of 0.1764 seconds, albeit with lower accuracy. TCNAtten followed with an execution time of 0.2114 seconds but outperformed all other models by achieving perfect accuracy (1.0). This performance advantage results from the synergy between the temporal convolutional network's efficient feature extraction capabilities and the multi-head attention mechanism's ability to model global dependencies. As a result, TCNAtten demonstrates both rapid responsiveness and high precision in processing time-series data related to the chilled water inflow of cooling units.

TABLE II
THE PARAMETERS OF TCNATTEN ARE AS FOLLOWS: I DENOTES THE NUMBER OF INPUT CHANNELS, O DENOTES THE NUMBER OF OUTPUT CHANNELS, K DENOTES THE KERNEL SIZE, S DENOTES THE STEP SIZE, SL DENOTES THE INPUT CHANNEL SIZE, D DENOTES THE DILATION RATE, AND P DENOTES THE PADDING AMOUNT

Path	layer	I	O	K	S	D	P
TCNAtten	tcn1	Sl	128	3	2	1	1
	tcn2	128	128	5	2	2	4
	tcn3	256	256	7	2	4	12

TABLE III
THE PARAMETERS OF HC2 ARE AS FOLLOWS: WHERE m IS THE SERIES LENGTH, d IS THE NUMBER OF DIMENSIONS AND rm IS THE LENGTHS OF DRCIF REPRESENTATIONS

Classification	Configuration
TDE	250 parameter sets sampled, 50 max ensemble size
DrCIF	500 trees, 4+(sqrt(rm)*sqrt(d))/3 intervals per representation, 10 attributes per tree, rm/2 max interval length
Arsenal	2,000 kernels per classifier, 25 ensemble size
STC	1 hour Shapelet Transform train time contract, 200 Rotation Forest trees

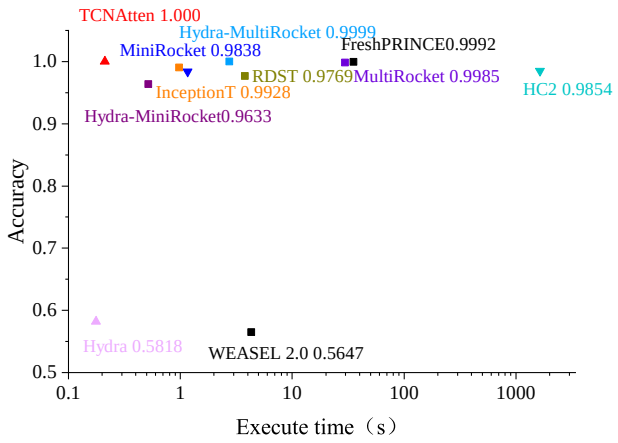

Fig. 7. Comparison of accuracy and implementation practices of the other models

V. ABLATION STUDY

To evaluate the impact of different model components on overall performance, we designed a series of ablation experiments focusing on removing or replacing key structures within the TCNAtten model. These components include adding a causal convolution structure to the TCN, removing the attention mechanism, and replacing the attention pooling layer with global average pooling. Unlike the previous chapter, which mainly focused on model accuracy and inference efficiency, this chapter introduces more comprehensive performance evaluation metrics that are more closely aligned

with real-world industrial applications—namely, Precision, Recall, F1 Score, and Mean Average Precision (mAP)—to assess the model's practicality and stability thoroughly. Since the TCNAtten model has already outperformed other baseline models regarding accuracy and inference efficiency compared to the previous chapter, this chapter does not re-evaluate those models. Instead, it focuses on exploring how different components influence the performance of TCNAtten.

A. Experimental Setup

Precision measures the model's ability to reduce false alarms in the chiller fault detection task, while Recall focuses on detecting whether faults are missed. The F1 Score balances the two, making it particularly suitable for scenarios with class imbalance. Mean Average Precision (mAP) evaluates detection performance across multiple fault categories, ensuring the model maintains balanced performance across different fault types. Therefore, we introduced Precision, Recall, F1 Score, and mAP in addition to Accuracy as evaluation metrics. Together, these metrics validate the model's practicality and reliability in industrial settings, particularly in mission-critical tasks where minimizing missed detections and false alarms is essential. To further investigate the impact of different architectural components on model performance, we conducted the following additional experiments based on the TCNAtten framework:

1) Adding causal convolution to the TCN (TCNAtten1): This experiment evaluates the effect of removing causal convolution on model performance.

2) Removing the attention mechanism (BaseTCN): This assesses the contribution of the attention mechanism to the model's capabilities.

3) Replacing the attention pooling layer with global average pooling (TCNAtten2): This investigates the impact of this structural change on performance.

B. Experimental Results

All experiments were conducted under the same conditions as those in the previous chapter. To minimize the impact of randomness, each configuration was run 30 times, and the average results were reported. Table IV and Table V present the overall performance comparison of different structural models and the comparison of average precision across categories, respectively.

TABLE IV
OVERALL PERFORMANCE COMPARISON OF DIFFERENT STRUCTURAL MODELS

Classifier	TCNAtten	TCNAtten1	BaseTCN	TCNAtten2
Accuracy	1.0000	0.9948	0.9997	1.0000
Precision	1.0000	0.9941	0.9999	1.0000
Recall	1.0000	0.9897	0.9995	1.0000
F1 Score	1.0000	0.9918	0.9997	1.0000
mAP	1.0000	0.9931	1.0000	1.0000
Time	0.22s	0.11s	0.07s	0.26s

Table IV shows that the TCNAtten model achieved a perfect score of 1.0 across all evaluation metrics. By incorporating a causal convolution structure into the TCN framework, the model significantly improved temporal efficiency—reducing inference time from 0.22 seconds to 0.11 seconds—while maintaining exceptionally high accuracy. This improvement can be attributed to causal convolution's ability to mask future time steps, thereby constraining information propagation and optimizing the computational pathway, making it more suitable for real-time industrial applications. However, this comes at the cost of a slight decline in precision and an increased false alarm rate. In contrast, the BaseTCN model, which entirely omits the attention mechanism, maintained an accuracy of 0.9997 but exhibited marginal declines in precision, recall, and F1-score. This highlights the critical role of the attention mechanism in enhancing the model's ability to identify anomalous signals at key time points. Notably, BaseTCN recorded the shortest inference time of just 0.07 seconds, making it ideal for scenarios demanding ultra-low latency but tolerating a higher margin of error. In experiments where attention pooling was replaced with global average pooling, overall performance remained largely unaffected; however, the model lost the dynamic temporal focus. In comparison, attention pooling improves recognition accuracy and provides superior interpretability. When combined with visualization tools, it enhances model transparency and maintainability.

TABLE V
COMPARISON OF AVERAGE PRECISION ACROSS CATEGORIES

Classifier	TCNAtten	TCNAtten1	BaseTCN	TCNAtten2
Class0	1.0000	0.9777	1.0000	1.0000
Class1	1.0000	0.9817	1.0000	1.0000
Class2	1.0000	0.9992	0.9989	1.0000
Class3	1.0000	0.9998	0.9999	1.0000
Class4	1.0000	0.9931	1.0000	1.0000
Class5	1.0000	1.0000	0.9996	1.0000

These ablation study results demonstrate that TCNAtten excels in accuracy, robustness, and interpretability. The integration of causal convolution effectively enhances real-time performance, while the attention mechanism proves indispensable for improving detection precision and capturing critical temporal segments. Although removing attention pooling does not drastically impair performance, it introduces latent deficits in interpretability and feature focus. Given that different industrial applications have varying accuracy, latency, and transparency requirements, the model architecture can be tailored and optimized accordingly to meet specific operational needs.

C. Model Analysis

The rapid responsiveness of TCNAtten is attributed to its TCN layers, particularly the implementation of dilated convolutions. By expanding the receptive field, dilated convolutions capture long-range temporal dependencies while maintaining low computational complexity, enabling TCNAtten to process large-scale data quickly. Moreover, integrating a multi-head attention mechanism further enhances its ability to model long-term dependencies, improving classification accuracy under complex patterns. In this study, TCNAtten accurately detects subtle fluctuations in traffic data and facilitates real-time assessment of device operating states. Ultimately, the model achieved a perfect score 1.0 across multiple key performance metrics, demonstrating superior classification accuracy and robustness.

1) Parameter Setting Recommendations and Efficiency-Accuracy Trade-offs in Cross-Domain Applications: When deploying the model in other domains (e.g., medical and health monitoring, energy management, and intelligent manufacturing), the parameters must be adaptively adjusted based on each specific scenario's data characteristics, task requirements, and hardware constraints. Below, we analyze the tuning strategies for key parameters and their impact on efficiency and accuracy.

Adjustment of the number of TCN layers: The number of TCN layers determines the model's ability to extract deep features from time series data. A greater number of layers expands the model's receptive field, enabling it to capture more complex temporal patterns but at the cost of increased computational complexity. For relatively simple tasks (e.g., temperature sensor monitoring), fewer TCN layers (e.g., 2–3 layers) are recommended to reduce computational overhead. Conversely, for more complex tasks (e.g., ECG signal classification or industrial equipment multi-fault diagnosis), increasing the number of layers to 4–6 can enhance the model's ability to capture long-range temporal dependencies. While adding more layers significantly enhances the model's capability to learn intricate patterns, it also increases training time and inference latency. Conversely, reducing the number of layers lowers computational demands but may compromise the model's ability to capture long-range dependencies.

Convolution kernel size and expansion coefficient settings: The size of the convolution kernel determines the coverage of the local time window. Larger convolution kernels capture a broader range of local features but increase the number of parameters. Expansion coefficients progressively expand the sensory field through exponential expansion (e.g., 1, 2, 4, 8, ...) to cover a longer period. For short-term data (e.g., heart rate monitoring), smaller convolution kernels (3–5) and moderate expansion coefficients (1–4) are recommended. For long-term data (e.g., energy consumption trends), larger convolution kernels (5–7) and exponentially growing expansion coefficients (1, 2, 4, 8...) are recommended. Increasing the convolution kernel size or expansion coefficients improves the ability to model long-range dependencies and increases memory usage and computation time. Conversely, decreasing these parameters reduces computational complexity but may sacrifice the ability to capture important temporal patterns.

Adjustment of Dropout Rate: The dropout rate suppresses overfitting by randomly discarding neurons, and the higher its value, the stronger the regularization effect. For small-scale datasets (e.g., medical diagnostic data), a higher dropout rate (0.4–0.6) is recommended to enhance generalization capability, while for large-scale or low-noise data (e.g., industrial sensor data), the dropout rate can be reduced (0.2–0.3) to retain more feature information. Increasing the dropout rate mitigates overfitting but may prolong convergence time; conversely, decreasing the dropout rate accelerates training but may lead to performance degradation on the test set due to overfitting.

Adjustment of the number of attentional heads: Multi-head attention mechanisms capture multi-dimensional dependencies through parallel subspaces, and the higher the number of heads, the higher the model complexity. Fewer attention heads (4–6) are suitable for low-dimensional data (e.g., single sensor signals), whereas higher-dimensional data (e.g., multimodal industrial data) require an increased number of heads (8–12) to extract features adequately. Increasing the number of heads improves global modelling capability but significantly increases computational resource consumption; conversely, decreasing the number of heads reduces the memory footprint but may ignore some key temporal relationships.

Optimizer and learning rate selection: Optimizers (e.g., Adam, SGD) and learning rate directly affect the model's convergence speed and stability. The Adam optimizer (with an initial learning rate of 0.001) is recommended for its dynamic tuning properties that balance convergence speed and stability. For highly noisy data, weight decay (e.g., 1e-4) or a reduced learning rate (e.g., 0.0005) can be combined to enhance robustness. Decreasing the learning rate helps improve training stability but may prolong convergence time; conversely, increasing the learning rate accelerates training but tends to trigger gradient oscillations or explosions.

The key parameters can be gradually adjusted based on the original paper's parameters in practical applications. The number of TCN layers and attention heads can be reduced for high real-time demand. For high precision requirements, the number of TCN layers and attention heads can be increased appropriately, and the dropout rate can be adjusted to prevent overfitting. Parameters must be dynamically adjusted according to data characteristics (e.g., noise level, time-series length, dimensionality) and task requirements (real-time performance, accuracy).

2) Model Limitations and Future Improvements: Although TCNAtten performs well in the chiller inlet flow task, the model may experience performance fluctuations when dealing with noisy or complexly varying time-series data. More accurate anomaly detection mechanisms, such as autoencoders or generative adversarial networks, could be introduced to enhance data cleaning and noise filtering. Furthermore, while TCNAtten effectively handles long sequences, it may face challenges related to computational complexity and capturing long-range dependencies in very long sequences. Multilayer inflationary convolution and adaptive time windowing strategies could be explored to further improve performance in long-sequence tasks. Regarding generalization, TCNAtten relies heavily on training with specific datasets and may underperform when encountering new data types. Therefore, incorporating cross-domain learning or meta-learning strategies would enhance the model's robustness and adaptability. Finally, while TCNAtten excels in responsiveness, a trade-off between real-time performance and accuracy in high-load scenarios may exist. The computational effort could be reduced by employing lightweight models or pruning techniques while maintaining high accuracy to optimize the model's performance under such conditions.

VI. Conclusion

This study proposes a real-time monitoring method for chiller cooling water inlet flow based on TSC. By analyzing the trend of the cooling water inlet flow in real-time, the method can promptly detect abnormal fluctuations and issue warnings about potential failures or abnormal conditions, thus significantly enhancing the reliability and stability of the system and preventing production interruptions and safety hazards caused by equipment failure. The method effectively

adapts to dynamic changes in the data, demonstrating excellent flexibility and wide applicability, and offers rapid response capabilities, enabling real-time decision-making. This makes the method particularly well-suited for application scenarios, such as chillers, where real-time monitoring is crucial. Additionally, the method is not limited to the chiller industry and has significant potential for cross-domain applications. In today's era of big data, it can be widely applied to other fields requiring real-time analysis, such as industrial equipment failure monitoring, intelligent manufacturing, energy management, and healthcare monitoring. This broad applicability highlights its substantial prospects and practical value.

REFERENCES

[1] M. Soori, B. Arezoo, and R. Dastres, "Internet of things for smart factories in industry 4.0, a review," *Internet of Things Cyber-Physical Systems*, vol. 3, pp. 192–204, 2023.

[2] L. Rosafalco, A. Manzoni, S. Mariani, and A. Corigliano, "Fully convolutional networks for structural health monitoring through multivariate time series classification," *Advanced Modeling Simulation in Engineering Sciences*, vol. 7, no. 1, p. 38, 2020.

[3] K. Gundersen, G. Alendal, A. Oleynik, and N. Blaser, "Binary time series classification with bayesian convolutional neural networks when monitoring for marine gas discharges," *Algorithms*, vol. 13, no. 6, p. 145, 2020.

[4] E. Burnaev, "Time-series classification for industrial applications: Road surface damage detection use case," *Journal of Communications Technology Electronics*, vol. 65, pp. 1491–1498, 2020.

[5] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the national academy of sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[6] K. Fukushima, "Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biological cybernetics*, vol. 36, no. 4, pp. 193–202, 1980.

[7] S. Bai, J. Z. Kolter, and V. Koltun, "An empirical evaluation of generic convolutional and recurrent networks for sequence modeling," *arXiv preprint arXiv:1803.01271*, vol. 10, 2018.

[8] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, vol. 30, 2017, p. 6000–6010.

[9] M. Middlehurst, J. Large, M. Flynn, J. Lines, A. Bostrom, and A. Bagnall, "Hive-cote 2.0: a new meta ensemble for time series classification," *Machine Learning*, vol. 110, no. 11-12, pp. 3211–3243, 2021.

[10] A. Dempster, D. F. Schmidt, and G. I. Webb, "Minirocket: A very fast (almost) deterministic transform for time series classification," in *Proceedings of the 27th ACM SIGKDD conference on knowledge discovery and data mining*, 2021, pp. 248–257.

[11] S. Alonso, A. Moran, D. Pérez, M. A. Prada, I. Diaz, M. Dominguez, and Applications, "Estimating cooling production and monitoring efficiency in chillers using a soft sensor," *Neural Computing*, vol. 32, no. 23, pp. 17 291–17 308, 2020.

[12] Q. Deng, Z. Chen, P. Tang, T. Peng, and C. Yang, "Performance monitoring and trend visualization of chillers based on principal component analysis and its application," in *2023 CAA Symposium on Fault Detection, Supervision and Safety for Technical Processes (SAFEPROCESS)*, 2023, pp. 1–6.

[13] C. Ji, Y. Wei, and X. Zheng, "Shapelet selection for time series classification," *Applied Soft Computing*, vol. 167, p. 112431, 2024.

[14] X.-M. Le, L. Luo, U. Aickelin, and M.-T. Tran, "Shapeformer: Shapelet transformer for multivariate time series classification," in *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2024, pp. 1484–1494.

[15] H.-Y. Liu, Z.-Z. Gao, Z.-H. Wang, and Y.-H. Deng, "Time series classification with shapelet and canonical features," *Applied Sciences*, vol. 12, no. 17, p. 8685, 2022.

[16] S. Jing and J. Yang, "Method of shapelet discovery for time series ordinal classification," *Soft Computing*, vol. 28, pp. 1–17, 2024.

[17] Y. Liu, Y.-A. Zhang, M. Zeng, and J. Zhao, "A novel distance measure based on dynamic time warping to improve time series classification," *Information Sciences*, vol. 656, p. 119921, 2024.

[18] T. Hayashi, D. Cimr, F. Studnička, H. Fujita, D. Bušovský, R. Cimler, and A. Selamat, "Distance-based one-class time-series classification approach using local cluster balance," *Expert Systems with Applications*, vol. 235, p. 121201, 2024.

[19] C. Wang, S.-j. Zhao, Z.-q. Ren, and Q. Long, "Place-centered bus accessibility time series classification with floating car data: An actual isochrone and dynamic time warping distance-based k-medoids method," *ISPRS International Journal of Geo-Information*, vol. 12, no. 7, p. 285, 2023.

[20] J. Large, A. Bagnall, S. Malinowski, and R. Tavenard, "On time series classification with dictionary-based classifiers," *Intelligent Data Analysis*, vol. 23, no. 5, pp. 1073–1089, 2019.

[21] R. Ayllón-Gavilán, D. Guijo-Rubio, P. A. Gutiérrez, and C. Hervás-Martínez, "A dictionary-based approach to time series ordinal classification," in *International Work-Conference on Artificial Neural Networks*, 2023, pp. 541–552.

[22] B. Bai, G. Li, S. Wang, Z. Wu, and W. Yan, "Time series classification based on multi-feature dictionary representation and ensemble learning," *Expert Systems with Applications*, vol. 169, p. 114162, 2021.

[23] C. Ji, M. Du, Y. Hu, S. Liu, L. Pan, and X. Zheng, "Time series classification based on temporal features," *Applied Soft Computing*, vol. 128, p. 109494, 2022.

[24] T. Iqbal, A. Elahi, W. Wijns, B. Amin, and A. Shahzad, "Improved stress classification using automatic feature selection from heart rate and respiratory rate time signals," *Applied Sciences*, vol. 13, no. 5, p. 2950, 2023.

[25] F. Karim, S. Majumdar, H. Darabi, and S. Harford, "Multivariate lstm-fcns for time series classification," *Neural networks*, vol. 116, pp. 237–245, 2019.

[26] R. Rahimilarki, Z. Gao, N. Jin, and A. Zhang, "Convolutional neural network fault classification based on time-series analysis for benchmark wind turbine machine," *Renewable Energy*, vol. 185, pp. 916–931, 2022.

[27] C. Yang, X. Wang, L. Yao, G. Long, and G. Xu, "Dyformer: A dynamic transformer-based architecture for multivariate time series classification," *Information Sciences*, vol. 656, p. 119881, 2024.

[28] Y. Wang, N. Huang, T. Li, Y. Yan, and X. Zhang, "Medformer: A multi-granularity patching transformer for medical time-series classification," *arXiv preprint arXiv:.19363*, 2024.

[29] M. Middlehurst, J. Large, G. Cawley, and A. Bagnall, "The temporal dictionary ensemble (tde) classifier for time series classification," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2020*, 2021, pp. 660–676.

[30] A. Dempster and F. Petitjean, "Rocket: Exceptionally fast and accurate time series classification using random convolutional kernels," *Data Mining and Knowledge Discovery*, vol. 34, no. 5, pp. 1454–1495, 2020.

[31] A. Bostrom and A. Bagnall, "Binary shapelet transform for multiclass time series classification," in *Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXII: Special Issue on Big Data Analytics Knowledge Discovery*, 2017, pp. 24–46.

[32] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, and F. Petitjean, "Inceptiontime: Finding alexnet for time series classification," *Data Mining Knowledge Discovery*, vol. 34, no. 6, pp. 1936–1962, 2020.

[33] M. Middlehurst and A. Bagnall, "The freshprince: A simple transformation based pipeline time series classifier," in *International Conference on Pattern Recognition and Artificial Intelligence*, 2022, pp. 150–161.

[34] C. W. Tan, A. Dempster, C. Bergmeir, and G. I. Webb, "Multirocket: Multiple pooling operators and transformations for fast and effective time series classification," *Data Mining and Knowledge Discovery*, vol. 36, no. 5, pp. 1623–1646, 2022.

[35] A. Dempster, D. F. Schmidt, and G. I. Webb, "Hydra: Competing convolutional kernels for fast and accurate time series classification," *Data Mining and Knowledge Discovery*, vol. 37, no. 5, pp. 1779–1805, 2023.

[36] A. Guillaume, C. Vrain, and W. Elloumi, "Random dilated shapelet transform: A new approach for time series shapelets," in *International Conference on Pattern Recognition and Artificial Intelligence*, 2022, pp. 653–664.

[37] P. Schäfer and U. Leser, "Weasel 2.0–a random dilated dictionary transform for fast, accurate and memory constrained time series classification," in *arXiv preprint arXiv:10194*, 2023.