IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 23, 2021, accepted September 28, 2021, date of publication October 13, 2021,
date of current version October 20, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3119573*

# E-SFD: Explainable Sensor Fault Detection in the ICS Anomaly Detection System

**CHANWOONG HWANG AND TAEJIN LEE**
Department of Information Security, Hoseo University, Asan 31499, South Korea

Corresponding author: Taejin Lee (kinjecs0@gmail.com)

This work is the result of commissioned research project supported by the affiliated institute of Electronics and Telecommunications Research Institute (ETRI) [2021-062].

**ABSTRACT** Industrial Control Systems (ICS) are evolving into smart environments with increased interconnectivity by being connected to the Internet. These changes increase the likelihood of security vulnerabilities and accidents. As the risk of cyberattacks on ICS has increased, various anomaly detection studies are being conducted to detect abnormal situations in industrial processes. However, anomaly detection in ICS suffers from numerous false alarms. When false alarms occur, multiple sensors need to be checked, which is impractical. In this study, when an anomaly is detected, sensors displaying abnormal behavior are visually presented through XAI-based analysis to support quick practical actions and operations. Anomaly Detection has designed and applied better anomaly detection technology than the first prize at HAICon2020, an ICS security threat detection AI contest hosted by the National Security Research Institute last year, and explains the anomalies detected in its model. To the best of our knowledge, our work is at the forefront of explainable anomaly detection research in ICS. Therefore, it is expected to increase the utilization of anomaly detection technology in ICS.

**INDEX TERMS** Explainable anomaly detection, ICS, HAI dataset, Bi-LSTM, XAI, SHAP.

## I. INTRODUCTION

An industrial control system (ICS) is a system that monitors and controls the state of geographically dispersed remote facilities for major national infrastructure providers, such as energy, gas, water, and traffic, in real time. The Programmable Logic Controller (PLC), the main element of the control system, is not connected to the internet, so there are few threats other than physical obstruction or natural disasters. Historically, these systems ran on proprietary hardware and software in physically secure locations, but in recent times they have adopted common information technology (IT) technologies and remote connectivity. Advances in smart sensors, Internet of Things (IoT), and wireless networks have been integrated with Operational Technology (OT) and IT for leveraging a high-speed, real-time response and ensuring cost-effectiveness. The arrival of new technologies like Virtualization, Cloud Computing, Software Defined Networks, Big Data Analytics, IoT, Machine Learning, and Artificial Intelligence have led to immense improvement in industrial productivity and system functions. These changes increase the likelihood of cybersecurity vulnerabilities and incidents [1], [2]. As the control system directly controls the on-site facilities, if a cyber-intrusion accident targets the control system, it can cause physical and economic damage as well as personal damage.

In 2010, Stuxnet, a malicious code targeting a control system, physically destroyed 1,000 centrifuges in Iran's nuclear facility. This proves that even closed/independent networks are not safe from cyberattacks and that cyberattacks can result in physical damage. In 2016, a massive blackout occurred owing to a cyberattack on the Ukrainian power grid [3]. It was proven that this incident could lead to a massive blackout. In 2017, an older attack was detected while preparing for attacks using the zero-day vulnerability of the firmware through the Engineering WorkStation (EWS) of a chemical plant in Saudi Arabia. This case had remained undetected for more than 3 years after it penetrated the IT network in 2014. After Stuxnet, interest in ICS security has increased, and various AI-based anomaly detection studies have attempted to ensure cyber safety not only in national infrastructure but also in ICS.

AI-based anomaly detection technology usually provides satisfactory prediction results. However, the biggest challenge with current AI is that experts cannot interpret the reasons for AI prediction results. This has to do with the

The associate editor coordinating the review of this manuscript and approving it for publication was Xiaochun Cheng.

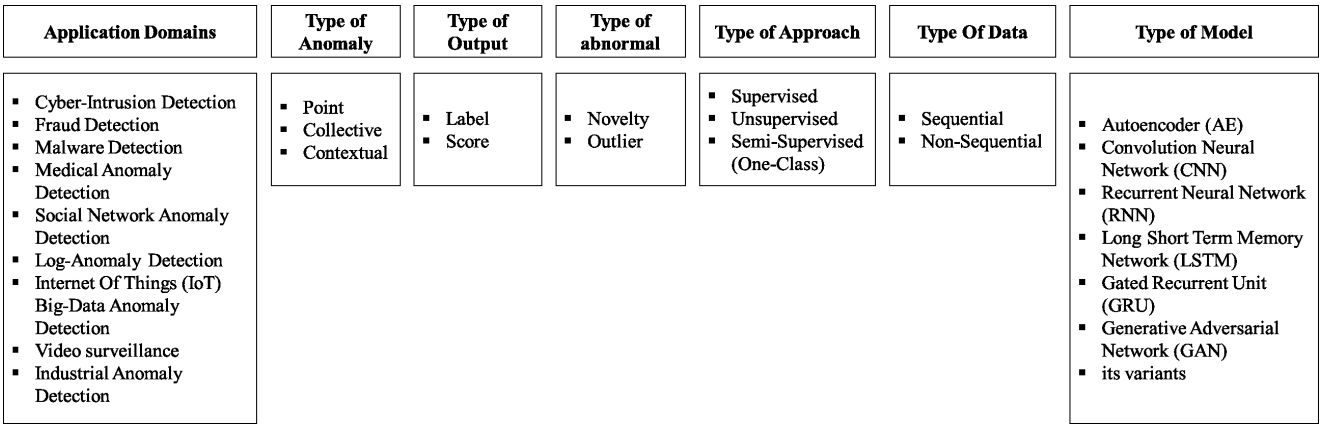


**FIGURE 1.** Key components related to deep learning-based anomaly detection technology.

reliability of AI models. Interpretation of the model is important because even if the model is highly accurate, it is not 100% reliable and can lead to false detection. Accurate analysis of interpretations is essential, so in high-security environments, security analysts currently need to intervene and respond to threats directly. Therefore, this study provides an explanation and interpretation of the abnormal situation predicted by the anomaly detection model. Security experts can analyze the features with the highest error for each feature, but our approach adds a layer that calculates the mean error for all features to identify the most important features that contributed to the anomaly score. It then aims to provide a comprehensive explanation to security experts by visualizing the features that contributed to the anomaly prediction. Using this visualization, security experts can establish a safe ICS environment by checking the sensors of corresponding features.

The major contributions of this paper are as follows:

- A better performance than the best team at HAICon2020 is obtained using our method.
- A layer is added that outputs the mean prediction error for all features so as to identify the features that contributed to the anomaly score without retraining the model.
- An interpretation of sensor fault detection is provided by visualizing the features that contributed to anomaly detection.

The rest of the paper is organized as follows. Section II presents some background on HAI dataset-based HAICon2020, anomaly detection, and XAI. Section III presents related work on anomaly detection and XAI. Section IV presents several models for anomaly detection that minimize false detections. Section V presents an approach for interpreting anomaly detection models. Section VI presents the evaluation results using multiple anomaly detection models and discusses the explainable sensor fault detection results. Section VII concludes this paper.

## II. BACKGROUND

### A. AI-BASED ANOMALY DETECTION TRENDS

One way to detect anomalous behavior of systems at the physical level is to leverage existing IT network-based intrusion detection systems (IDS) to identify malicious activity. Anomaly detection, a type of IDS, is a critical data analysis task that detects anomalies or abnormalities in a given dataset. It is an interesting area of data mining research that involves identifying rare patterns that deviate from normal behavior. Anomaly detection has been studied for a long time using statistics, and various detection techniques, including machine learning and deep learning, are currently being studied. These AI technologies are increasingly being used to automate anomaly detection. Figure 1 shows the major components involved in deep learning-based anomaly detection technology. Anomaly detection is being used in various application domains [4], [5]. Anomaly detection seems very simple but involves looking for data that does not follow normal behavior patterns. Unfortunately, creating a precise model of complex physical processes is very challenging. Despite the many technologies available, there are several research challenges:

(1) Imbalanced Data: In the case of supervised approaches wherein all labels exist in the dataset, the more data there is, the better the performance. However, when generating millions of samples in practice, anomalies may occur only rarely.

(2) Data Labeling: In order to obtain anomaly data, it is necessary to perform labeling through expert analysis. This costs a considerable amount of time and money.

(3) Unknown anomalies: Various types of anomalies may occur rather than just a few known fixed forms.

The field of AI-based anomaly detection is broadly classified into supervised, semi-supervised, and unsupervised detection based on the presence or absence of labels. Supervised approaches require prior labeling of system behavior, including malicious behavior samples. Collecting accurate and representative labeled data is actually very difficult,

and such data is highly system-dependent. Therefore, the need for unsupervised and semi-supervised approaches has emerged [6], [7]. Unsupervised Supervisory Control and Data Acquisition (SCADA) intrusion detection were investigated in a previous study [8], which explains a technique based on single-class SVM and k-means clustering. The performance is not as high as that of the supervised approach; it is very sensitive to hyper-parameters. Semi-supervised (or one-class classification) is an approach to machine learning that combines a small amount of labeled data with a large amount of unlabeled data during training. Semi-supervised learning falls between unsupervised (with no labeled training data) and supervised (with only labeled training data). It is a special instance of weak supervision. Therefore, semi-supervised approaches to anomaly detection aim to utilize some labeled samples, but most proposed methods are limited to merely including labeled normal samples. While both approaches should be examined closely, they can be fulfilled in many practical situations. Semi-supervised methods generally have lower false positive rates than fully unsupervised methods [9], [10].

Anomaly detection can be approached in various ways depending on the type of data. As most anomaly detection datasets are sequential, an anomaly detection model that adequately reflects the characteristics of the data must be chosen. These workflows do not deviate from the following categories. First, the input is defined in subsequence units in time-series data and future data is predicted. Second, only normal data is trained using statistics, machine learning and deep learning, and other relevant techniques. Finally, if the error between the predicted value of the detection model and the actual value is higher than the threshold, it is regarded as abnormal; otherwise, it is normal. However, in terms of ICS operation, it is very difficult to create a precise model because it is difficult to guarantee availability when the process is stopped owing to false detection. Therefore, the anomaly detection approach in this study creates a model for each process and each time step model, and detects clear attacks by setting a threshold to minimize false detections in each model.

### B. HIL-BASED AUGMENTED ICS SECURITY DATASET (HAI DATASET)

The HAI dataset was collected from a realistic ICS testbed augmented with a Hardware-In-the-Loop (HIL) simulator that emulates steam-turbine power generation and pumped-storage hydropower generation. A detailed description of the test bed and dataset can be found in [11], [12], but a brief description is provided below. The process flow of the testbed is divided into four main processes as shown in Figure 2: boiler process (P1), turbine process (P2), water-treatment process (P3), and HIL simulation (P4). The HIL simulation enhances the correlation between the three real-world processes at the signal level by simulating thermal power generation and pumped-storage hydropower generation scenarios. The boiler and turbine processes were used to

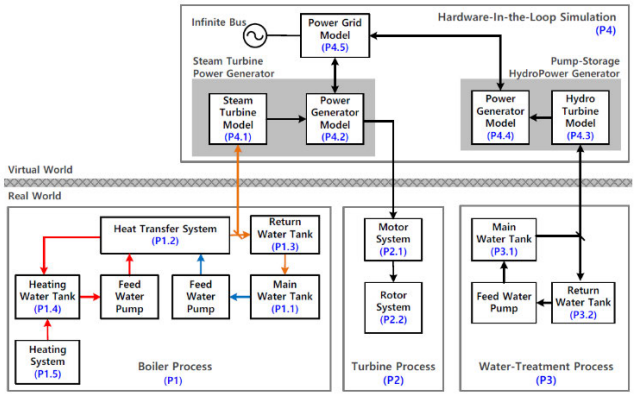


**FIGURE 2.** Process flow diagram.

simulate the thermal power plant, while the water treatment process was used to simulate the pumped-storage hydropower plant. Each of these processes involves multiple PLCs that can be managed by a single Human Machine Interface (HMI). An individual PLC is composed of Process Variable (PV), Set Point (SP), and Control Variable (CV), and is used in both SCADA and Distributed Control Systems (DCS) as a control device for automatic control and monitoring through feedback control. The flow of the process interprets the PV collected through sensors such as pressure, temperature, and speed, and the controller generates a CV and sends it to actuators such as valves and motors. The HMI for integrated ICS control monitors multiple PLCs and has a structure that can change the SP according to the process procedure [13]. The ICS operation structure is shown in Figure 3.

The HAI dataset contains data collected every second from all sensors and actuators, and this data was used for training and testing in our study. This dataset has been released in two major versions, and the configuration is summarized in Table 1. In this study, the dataset used for explainable sensor fault detection is the HAI 21.03 dataset. The most recent version of HAI 21.03 satisfies time continuity and includes 84 columns. The first column represents the observed time and the following 78 columns provide the recorded SCADA data points. The last four columns provide data labels for whether an attack occurred or not, where the attack column was applicable to all processes and the other three columns (i.e., P1, P2, and P3) were for the corresponding control

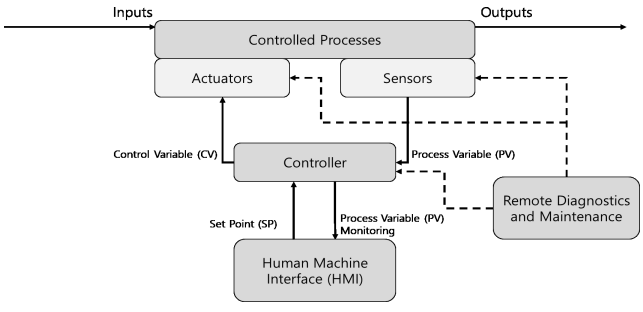


**FIGURE 3.** ICS operation structure.

**TABLE 1.** HAI dataset.

| Version | Data Points | Normal | | Attack | | |
|---|---|---|---|---|---|---|
| | | Interval | Size | Attack Count | Interval | Size |
| HAI 21.03 | 78 point/sec | 352 hours | 462 MB | 50 attacks | 112 hours | 208 MB |
| HAI 20.07 | 59 point/sec | 177 hours | 255 MB | 38 attacks | 123 hours | 181 MB |

processes. An HMI operation task scheduler was used to periodically set the SPs and HIL simulator variables to predefined values within the normal range to simulate a benign scenario. Fifty attacks were conducted, including 25 attack primitives and 25 combinations of attacks designed to simultaneously perform two attack primitives.

### C. AI-BASED ICS ANOMALY DETECTION CONTEST ON HAI DATASETS (HAICon2020)

HAICon2020 was a competition that judged the anomaly detection performance of a semi-supervised learning model for detecting abnormal (or unknown) behaviors that do not appear in the training data under normal conditions. A detailed description of HAICon2020 is presented in [14], but a brief description is provided below. Participants were provided with datasets (raw version of HAI 21.03), a baseline model, and the evaluation tool eTaPR (enhanced version of TaPR [15]) to help them understand how the competition would be conducted. For the baseline model, the RNN-based predictive model was trained after data preprocessing was performed to remove high-frequency noise on the training data. If the difference between the predicted results through the prediction model and the actual value exceeded a predetermined threshold, it was judged as an abnormal situation. When participants submitted a detection result file through the competition website, public and private eTaPR scores were automatically calculated. The public score was scored at about 30% of the total test data and was posted on the leaderboard during the competition. The private score was scored with the rest of the test data and released immediately after the competition ends. HAICon2020 was ranked by the final private score. A total of 890 teams participated, and the final scores of the top teams are listed in Table 2. The baseline model ranks 156th, but the top seven teams submitted anomaly detection models developed based on the baseline model. Most of the data preprocessing removes unchanged functions and uses normalization techniques. In data post-processing, anomaly scores were normalized and thresholds for judging abnormal situations were adjusted. The learning model seems to have produced results by changing various parameters. It would have been more advantageous to improve the performance through baseline model optimization rather than developing a new model, given the short competition period of approximately two months. Obviously, even a very simple model based on RNN achieves good performance. It seems that data pre-processing and post-processing of prediction errors have substantial influence on the detection performance improvement as compared to the learning model. The results of the top teams in the competition can be used as a performance comparison criterion when using HAI 21.03.

**TABLE 2.** Scores and ranking of top seven teams in HAICon2020.

| Final ranking | eTaPR Score (Ranking) | | |
|---|---|---|---|
| | Public | Private | Reproduction |
| 1 | 0.9799 (03) | 0.9379 (01) | 0.9378 (01) |
| 2 | 0.9803 (02) | 0.9361 (02) | 0.9363 (02) |
| 3 | 0.9701 (20) | 0.9275 (04) | 0.9275 (03) |
| 4 | 0.9741 (11) | 0.9341 (03) | 0.9239 (04) |
| 5 | 0.9751 (09) | 0.9235 (06) | 0.9197 (05) |
| 6 | 0.9765 (08) | 0.9216 (07) | 0.9186 (06) |
| 7 | 0.9698 (22) | 0.9175 (10) | 0.9125 (07) |
| 156 (Baseline) | 0.9226 (162) | 0.8416 (156) | - |

### D. EXPLAINABLE ARTIFICIAL INTELLIGENCE (XAI)

Neural networks are a structure in which millions of complexly connected parameters interact nonlinearly. Their performance is significantly higher than that of conventional machine learning. However, they are expressed as black boxes because even the developer cannot know exactly why the AI produced a given result, owing to the complexity of the internal structure. In recent times, XAI, which helps humans understand the reasons for AI-predicted results, has emerged [16], [17]. XAI is a technology that can explain the process by which results are generated so that humans can understand and interpret the results predicted by an AI model. According to Defense Advanced Research Projects Agency (DARPA) [18], XAI aims to produce more explainable models, while maintaining a high level of learning performance (prediction accuracy). It also aims to enable human users to understand, appropriately trust, and effectively manage the emerging generation of artificially intelligent partners. However, XAI is more difficult to interpret when the model is more complex. Conversely, the simpler the model, the easier it is to interpret. Thus, the complexity and interpretability of the model have a trade-off relationship with each other.

XAI interpretation methods can be classified into global and local depending on the scope of explaining the model. A global method is an interpretation method that always explains all of the predicted results by the model, while a local method is an interpretation method that explains only one or some of the predicted results. Given that local methods have less scope for explanations than global methods, one or some of the predictions can be explained to near perfection, even if the overall predictions cannot be explained.

XAI can also be classified according to whether the explanation technique is applicable only to a specific kind of model or universally. An explanation technique that can only be applied to a specific type of model is called model-specific. For example, all visual interpretation techniques that can

only be used in the Convolutional Neural Network (CNN) series are model-specific (e.g. Deconvolution, CAM, and Grad-CAM). This has the disadvantage of reducing the number of options when choosing a model, because it can only be applied to a specific model. By contrast, an explanation technique that can be applied regardless of the model is called model-agnostic (e.g., Surrogate Analysis, Sensitivity Analysis). In this case, humans cannot know the inside of the model, so in order to explain the model, it is necessary to find evidence outside the model. As such, it is possible to increase the utilization of anomaly detection technology by presenting a reliable basis for AI decisions through XAI. Therefore, this study uses SHapley Additive exPlanations (SHAP), a model-agnostic technique, to interpret the contribution of features in a prediction model without degrading the prediction accuracy of a complex anomaly detection model.

## III. RELATED WORK

### A. ANOMALY DETECTION RESEARCH

Anomaly detection involves searching for data that does not follow the generally expected pattern in a specific domain or that has a different pattern than the normal stream. Within an anomaly detection method, the choice of deep neural network architecture mainly depends on the type of input data. Input data can be broadly classified as sequential or non-sequential data. Haselmann *et al.* [19] proposed image-based anomaly detection using Convolutional Neural Networks (CNN) for surface inspection used in the manufacturing industry. Kim *et al.* [20] used an unsupervised AutoEncoder (AE) to detect unknown attacks in a single event. Some studies detect abnormal behavior of malicious code using principal component analysis, similar to AE [21]. Zenati *et al.* [22] provided a promising approach to solving the data imbalance problem by modeling real data into complex high-dimensional distributions using Generative Adversarial Network (GAN). They proposed an anomaly detection model that derives adversarial learned features and used reconstruction errors based on it to determine if the data sample is abnormal. When data similar to real anomalies are generated and trained through GAN, the model is expected to stabilize, improving the anomaly detection performance.

With the recent increase in IoT devices, anomaly detection models use datasets collected sequentially from various application domains. Sequential data are not independent and reflect temporal features. Depending on the type of data, the architecture for anomaly detection is different, and an architecture that reflects the characteristics of the data should be adopted. Hwang *et al.* [23] compared the performance of a non-sequential AE architecture and a sequential 1D-CNN architecture for anomaly detection in an Endpoint Detection and Response environment. Sequential architecture models are important because they detect what non-sequential architecture models cannot. Kravchik and Shabtai [24] proposed an anomaly detection method using a Secure Water Treatment testbed (SWaT) dataset representing a scaled-down version of an actual industrial water treatment plant. This method detected 31 out of 36 different cyberattacks, which represents successful detection and improvement compared to previous non-sequential architecture studies. This suggests that the sequential 1D-CNN architecture is effective in time-series prediction tasks. Recurrent Neural Network (RNN) series, as well as 1D-CNN, are effective for sequential data. Yin *et al.* [25] proposed a deep learning approach for intrusion detection using RNN. This approach is also compared with machine learning methods proposed by previous researchers. The proposed RNN-IDS is well-suited for modeling classification models with high accuracy, and its performance has been confirmed as being superior to existing machine learning classification methods in binary as well as multi-class classification. Wang *et al.* [26] configured Long Short-Term Memory (LSTM) as an AE and used it as a complex model. The designed model can perform input reconstruction as well as prediction and use both simultaneously to train better data. Naseer *et al.* [27] created and compared IDS models using various machine learning and Deep Neural Network (DNN) architectures. For the experiment, the NSL-KDD dataset was used based on the data collected in the IDS evaluation program. In the DNN model, the LSTM model achieved the highest accuracy, followed by deep CNN and convolutional AE. It is thus demonstrated that CNN and LSTM have excellent performance as architectures for anomaly detection in sequential datasets. Table 3 lists examples of anomaly detection models used based on the type of data.

**TABLE 3. Example of anomaly detection models used based on the type of data.**

| Type of Data | Example | Anomaly Detection Model Architecture | Ref. |
|---|---|---|---|
| Non-Sequential | Image, Sensor, Network Event, Other (data) | CNN, AE, its variants | [19], [20], [21], [22] |
| Sequential | Video, Speech, Protein Sequence, Time Series, Text (Natural language) | CNN, RNN, LSTM, GRU | [23], [24], [25], [26], [27] |

### B. XAI RESEARCH

XAI evolved to explain the black box model. Although there are a variety of XAI technologies, they are usually designed for images and rely on visual interpretability to evaluate and provide explanations. XAI is at an early stage of research. There are attempts to apply it not only to images but also to time series [28]. Morichetta *et al.* [29] proposed LIME to explain the results of unsupervised clustering. LIME [30] is one of the surrogate analysis methods and does not make any assumptions about the behavior of the model because it is model-agnostic. Therefore, LIME is an XAI that can be applied regardless of how the model is trained. One of the XAI technologies, Layer-wise Relevance Propagation (LRP) calculates a relevance score to determine the extent to which

backpropagation activity is affected by the forward propagation method in deep learning. Patil *et al.* [31] designed an LSTM model for sequential anomaly detection and explained the prediction results using LRP. However, the LSTM model for anomaly detection is designed with a shallow structure, so anomaly detection performance improvement is required. Shrikumar *et al.* [32] proposed Deep Learning Important FeaTures (DeepLIFT), a method of decomposing the output prediction of a neural network for a specific input by backpropagating the contribution of all neurons in the network to all features of the input. DeepLIFT compared the activation of each neuron to its 'reference activation' and assigns contribution scores according to the difference. One of the most successful ways to generate a saliency map is using CAM and grad-CAM [33]. CAM and grad-CAM are some of the XAI techniques that operate based on CNNs. In the case of CAM, there is a limitation of designing and training the model again because the CNN must be changed to GAP instead of the last FC layer. Assaf *et al.* [34] provided an explanation of prediction using gradient-based grad-CAM to generate a saliency map for multivariate time series prediction. Related studies organized by XAI type are listed in Table 4. Although several studies related to XAI are in progress, they are still in the early stages and are difficult to apply to deep anomaly detection models.

**TABLE 4. Related work on XAI.**

| Type of Data | Type of XAI | Model Architecture | Ref. |
|---|---|---|---|
| Non-Sequential | LIME | Clustering | [29], [30] |
| Sequential | LRP | LSTM | [31] |
| Non-Sequential | DeepLIFT | DNN | [32] |
| Sequential | Saliency maps (CAM, Grad-CAM) | CNN | [33], [34] |

## IV. ANOMALY DETECTION APPROACH USING MULTIPLE MODELS

In this session, a bidirectional stackable LSTM-based (Bi-LSTM) anomaly detection model for detecting anomalies in ICS is proposed. When detecting an abnormal situation in an actual industrial control system environment, it is necessary to minimize false alarms. If a physical process is stopped because of false alarms, there is a cost owing to availability problems, but if false detections are relatively reduced, certain anomalies may go undetected. Therefore, it is difficult to guarantee good performance with a single model. The proposed approach sets a threshold to minimize false detection in five models for each time step and four models for each process, and if even one of these models predicts abnormalities, it is considered as a definite abnormal. This approach is an assumption to minimize false detections in each model and to detect a wider variety of anomalies collectively across all models.

### A. DATA NORMALIZATION

As the scale of features collected from each sensor and actuator is different, it can be biased towards a particular feature during model training. i.e., given that the value range of each feature is different, it is essential to perform preprocessing by normalizing each feature before training the model. Normalization is a technique that converts values so that all features used have a similar influence. The minimal and maximal values of features from the training set were saved and used for scaling the test set. The max-min regularization makes all features to scale [0, 1]. All features with the same minimum and maximum values are set to 0.

### B. DATA PREPROCESSING

The prediction mode of the proposed model is a regression model that predicts the next data when a series of sequential data is input. In order to train a recurrent neural network on multivariate time series data, it is necessary to sample a part of the dataset by setting the window size. Sampling involves generating input data of a three-dimensional shape (sample, time step, features) and output data of a two-dimensional shape (sample, features) through a sliding window from time-series data. For example, if you set the window size to 60 seconds, then 1 to 59 seconds are used for training and the 60th second is the value the model should predict (data in the HAI dataset were collected every second). As the anomaly detection results differ depending on the window size, it is empirically set from 56 to 60, so that five models can be trained. The stride parameter is the sliding size. If the stride parameter in the time series is 1, the previous and next samples are similar, but they are used for sophisticated training.

As mentioned in Session II. B, the HAI dataset consists of four processes, and the model can be trained by classifying

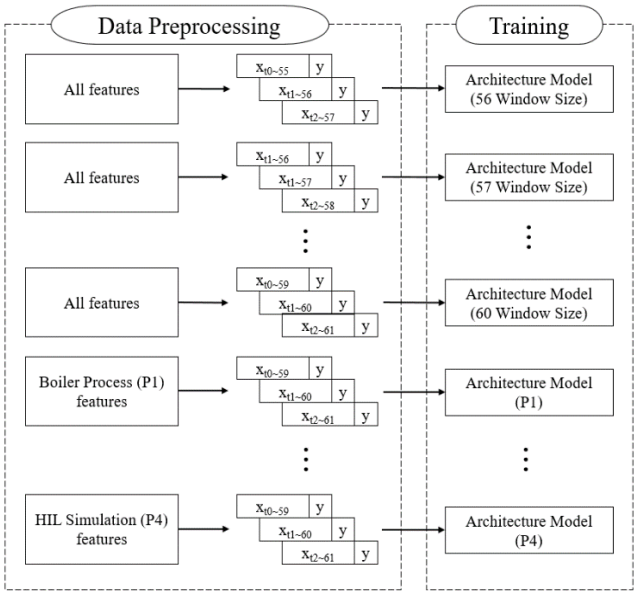


**FIGURE 4. Anomaly detection approaches using multiple models.**

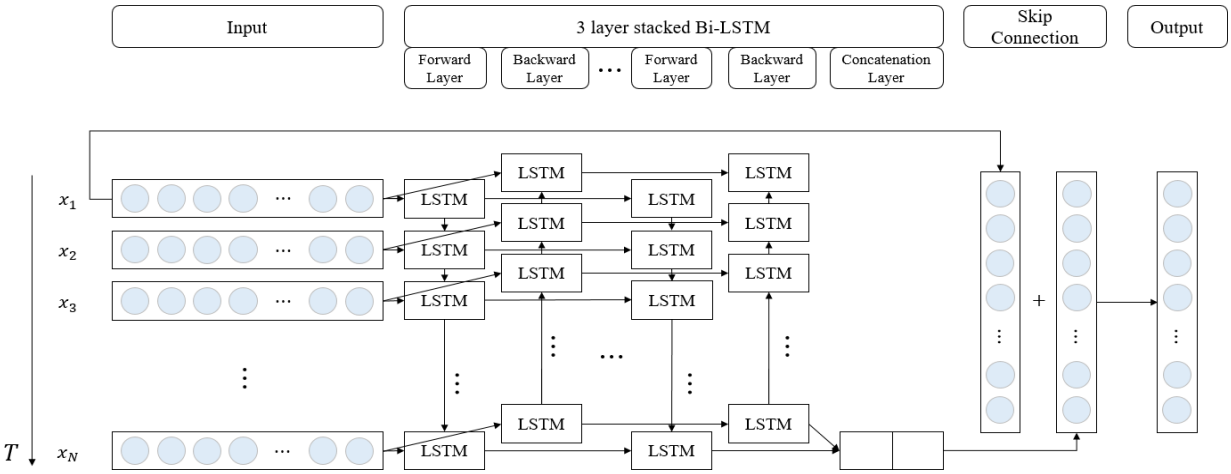


**FIGURE 5.** Architecture model.

the features of each process. Multiple models with each window size model and each process model consist of nine models as shown in Figure 4. Each window size model utilizes all features and each process model utilizes only the features of each process using 60 window sizes.

### C. ARCHITECTURE MODEL

In vanilla RNN, the output depends on the results of previous computations. However, vanilla RNN has the disadvantage that it is effective only for relatively short sequences. As the vanilla RNN time step increases, the input information cannot be sufficiently transmitted forward. This is called a long-term dependency problem. LSTM is a type of RNN that allows the network to retain long-term dependencies between data at a given time from many time steps before. The LSTM architecture reduces the long-term memory problem by using three gate units (i.e., the forget gate, the input gate, and the output gate) to control the memory of new information and the oblivion of past information. LSTMs can control how far past data is remembered by adding a cell state to the hidden state of an existing RNN. LSTM has been shown to perform well in sequence-based tasks with long-term dependencies compared to RNNs [35], [36]. Although a variety of LSTM variants were proposed in recent years, a comprehensive analysis of LSTM variants shows that none of the variants can improve upon the standard LSTM architecture significantly [37].

The architecture model used in this study adopts Bi-LSTM, which uses both forward and reverse directions simultaneously for precise learning of sequences, and is designed to solve more complex tasks by stacking them in three layers. Layers 1 and 2 have a many-to-many structure, while the last layer 3 is a many-to-one structure. For example, if an input x = *(1, 2, 3, 4, 5)* and output y = *(6)*, then the last layer has a many-to-one structure. Usually in neural networks, it is more efficient to stack hidden layers than to increase the number of hidden layer nodes to improve the model's performance. However, with more layers and deeper neural networks, the gradient vanishing problem increases. On models with dozens of layers, adding more layers does not improve performance. Therefore, our architectural model applies skip-connection (residual-connection). This method adds the first values of the input window to the output of the model, so that deeper layers can be trained and predicted well [38]. The designed Bi-LSTM model structure is shown in Figure 5.

### D. ANOMALY DETECTON METHOD

The detection criteria are based on the anomaly score calculated by the difference (error) between the actual value (i.e., the next value of the input that actually occurred) and the predicted value (i.e., next value of the input predicted by the model) [39]. The difference between the predicted value and the actual value is used as an anomaly score through the Mean Absolute Error (MAE). The formula for MAE is as follows:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{Y}_i - Y_i| \tag{1}$$

where $n$ is the number of predicted dimensions (the number of features), $\hat{Y}$ is the predicted value, and $Y$ is the actual value. The high anomaly score is based on the assumption that it is a pattern not previously seen in the training dataset.

In statistics, a Moving Average (MA) is a calculation used to analyze data points by creating a series of averages of different subsets of the complete dataset. In finance, a MA is a stock indicator that is commonly used in technical analysis. The reason for calculating the MA of a stock is to help smooth out the price data by creating a constantly updated average price. This moving average approach is applied to anomaly detection models. To minimize false detections, anomaly scores are equally weighted according to the following formula:

$$MA_t = \frac{1}{n}\sum_{i=1}^{n} p_{t-i} \tag{2}$$

where $n$ is the size of the subset and $p_t$ is the anomaly score at the prediction time $t$, i.e., MA is the mean of the previous $n$ values, excluding the anomaly score at the prediction time $t$. The final anomaly score is calculated as the average of the anomaly score and the $MA_t$ value at the prediction time $t$, and the formula is as follows.

$$AS_t = \frac{MA_t + p_t}{2} \tag{3}$$

finally, set the threshold value $T$. If

$$AS_t > T \tag{4}$$

then the state is regarded as abnormal; otherwise, it is normal.

### E. GRAY ZONE SMOOTHING

There is a sequence of each feature according to the operating structure of the industrial control system, e.g., when the temperature of the boiler sensor drops, water is pumped into the heat transfer system to operate the boiler, closing the valve and raising the water level in the water tank. Once a certain amount of water has been filled, the pump will stop working. In this process, the SP and CV change according to the usual PV change, and then the process of changing the PV is repeated. However, while SP and CV change, PV may increase relatively slowly or be delayed. Therefore, even if a model is well trained, false detection may occur at a certain point in time, as there may be some errors in collecting data every second from a sensor or actuator. When error data (1 second) is input to an anomaly detection model trained on normal data, it affects the preceding and succeeding time steps, and it is assumed that the predicted normal–abnormal sequence size will be higher than or similar to the time step. Therefore, the proposed model makes a final decision considering the predicted normal–abnormal sequence size to minimize these false detections. In an actual normal sequence, an anomaly score for a particular sequence is higher than the threshold and is considered an abnormal sequence. Conversely, in an actual abnormal sequence, the anomaly score of a specific sequence is lower than the threshold and is considered a normal sequence. Sequences that are difficult to consider normal or abnormal are called the gray zone. Figure 6 shows the gray zone in the actual normal sequence. To reduce gray areas (i.e., false detections), $n$ is set less than half the time step and the label is changed if the predicted normal–abnormal sequence size is less than $n$ seconds. If the predicted normal sequence is n seconds or less, it is considered abnormal, and if the predicted abnormal sequence is n seconds or less, it is considered normal. This is called gray zone smoothing and can minimize false alarm.

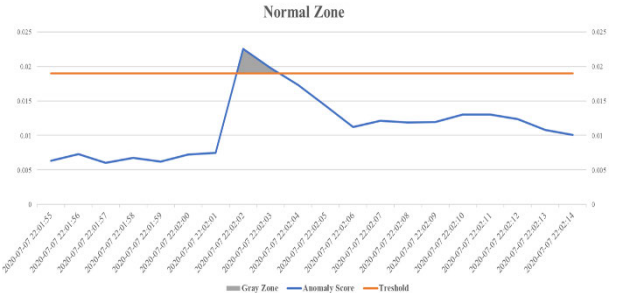


**FIGURE 6.** Example of a gray zone that is difficult to regard as an attack in the normal zone.

## V. EXPLAINABLE ANOMALY DETECTION APPROACH

The purpose of anomaly detection in industrial control systems is to prevent security accidents by detecting abnormal situations and responding quickly. However, although the current AI model guarantees high detection accuracy, its response is inadequate because it cannot provide an explanation for what the model has decided. This section proposes an explanation method that can effectively respond by checking the cause determined by the model using SHapley Additive exPlanations (SHAP), an XAI technique. SHAP provides interpretable predictions for test samples through the contribution score of each feature in a complex learning model. Our goal is to interpret the feature contribution of predictions using SHAP without compromising the accuracy of complex anomaly detection models.

### A. SHAPLEY ADDITIVE EXPLANATIONS

As important as the prediction of the model is the interpretation of the model. Several researchers have proposed applying machine learning methods that output an interpretable result from the prediction model. However, the existing interpretation approach employs simple models, such as linear regression and decision trees. The best explanation of a simple model is the model itself. It perfectly represents itself and is easy to understand. Complex models such as ensemble methods or deep networks are not easy to understand, so the original model cannot be used as the best explanation. Instead, the original model can use a simpler explanatory model that defines it as an interpretable approximation.

The SHAP framework [40] approach is shown in Figure 7. The main approach is to interpret it as a simple model $g(x)$ instead of a complex model $f(x)$. Given an input x, the simplified variable $x'$ for ease of interpretation is defined as a mapping function with $x = h_x(x')$. The surrogate model is trained with $g(z') \approx f(h_x(z')$ whenever $z' \approx x'$. The model $g(z')$ can be formulated as follows.

$$g\left(z'\right) = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i \tag{5}$$

where M is the number of simplified input features, $\phi_0 = f(h_x(0))$ is the model output without all of the simplified inputs, and $\phi_i \in \mathbb{R}$ represents the contribution of each feature to the model. Thus, an additive feature attribute can be used to create an explanatory model $g$ that satisfies $g(z) \approx f(h_x(z))$ through a simplified input $z'$, instead of the original input x. The additive feature attribute must satisfy all three properties:

1. Local accuracy: The explanation model $g\left(x'\right)$ has to match the output of the original model $f\ (x)$.

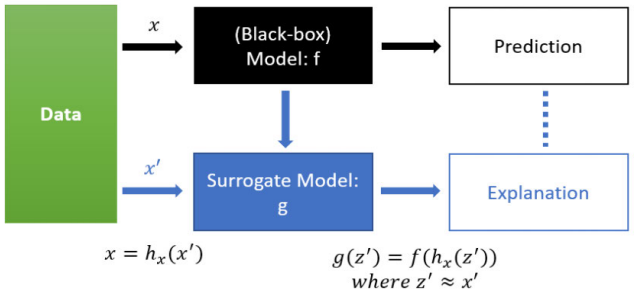


**FIGURE 7.** SHAP framework.

2. Missingness: Features missing in the original input must have no impact. i.e., $x'_i = 0 \Rightarrow \phi_i = 0$
3. Consistency: If we revise a model such that it depends more on a certain feature, then the contribution of that feature should not decrease, regardless of other features.

The only way to satisfy the definition of the additive feature attribute and the three desired properties is to use Shapley values. The main idea is to calculate the value through the average change depending on the presence or absence of a feature as a combination of several features for the importance of one feature. The contribution $\phi$ for each feature is known as the Shapley value and is calculated using the following equation.

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\,(|N| - |S| - 1)!}{|N|!} \left(v\left(S \cup \{i\}\right) - v\left(S\right)\right) \quad (6)$$

where $N$ is for all feature sets; $S$ is the set of all except the feature $i$; and $v\,(x)$ is the contribution of a subset of $x$. The average of the difference between the results of the model with and without feature $i$ is extracted as a measurement value. This is to measure the contribution of the model based on the presence or absence of feature $i$. This value is called the SHAP value, and is equivalent to the Shapley value in game theory [41]. Thus, SHAP is an algorithm that explains a model that satisfies the additional function attribute method using Shapley values.

### B. SHAP FRAMEWORK ACCORDING TO MODEL TYPE

SHAP is a unified approach to explain the result of any deep learning model, which connects the theory of games with local explanations, joining several methods and representing the additive feature attribution method. The SHAP framework can achieve fast calculation speed by various calculation methods depending on the type of model $\boldsymbol{f}$. There are model-agnostic approximation methods (e.g., Kernel SHAP) and model-specific approximation methods (e.g., Tree SHAP, Deep SHAP, and Gradient SHAP) [42].

Kernel SHAP is a method that uses a special weighted linear regression to compute the importance of each feature. The computed importance values are Shapley values from game theory and also coefficients from a local linear regression. Kernel SHAP could not be computationally configured for an LSTM model with three dimensions as inputs because it needs one or two dimensions passed through the prediction model, i.e., if the input data is designed as single value rather than sequential values, Kernel SHAP can be used because it provides a specific matrix structure in the form of two dimensions. From this point of view, the kernel SHAP is model-agnostic. However, it is slow and cannot be used if Shapley values need to be calculated for many instances.

By restricting ourselves to specific model types, such as Tree SHAP, Deep SHAP, and Gradient SHAP, faster approximation methods are obtained. Tree SHAP is a fast and accurate algorithm that calculates SHAP values for trees (e.g., decision tree) and tree ensembles (e.g., XGBoost, LightGBM). Deep SHAP and Gradient SHAP approximate SHAP values in deep learning models. Deep SHAP is a high-speed approximation algorithm for the SHAP values of deep learning models that builds on a connection with DeepLIFT. The implementation here differs from the original DeepLIFT by using a distribution of background samples instead of a single reference value and by using Shapley equations to linearize components such as max, softmax, products, and divisions. Note that some of these enhancements have also been since integrated into DeepLIFT. Gradient SHAP explains the model using the expected gradient (an extension of integrated gradient). The expected gradients method is an extension of the integrated gradients method [43], a feature attribution method designed for differentiable models based on an extension of Shapley values to infinite player games (Aumann–Shapley values). Integrated gradients values are slightly different from SHAP values and require a single reference value to integrate from. Expected gradients, also known as Gradient SHAP, arose as an adaptation to make these methods approximate Shapley values. As an adaptation to make them approximate SHAP values, the expected gradients method reformulates the integral as an expectation and combines that expectation with sampling reference values from the background dataset. This leads to a single combined expectation of gradients that converges to attributions that sum up the difference between the expected model output and the current output.

### C. EXPERIMENTAL DESIGN

Two explanation methods have been introduced as a SHAP framework (Deep SHAP and Gradient SHAP) for explaining deep learning models. We attempted to decide which of them is the most suitable to explain our data and model. Deep SHAP has a version compatibility issue that prevents it from working with Python 3 and TensorFlow 2+, and it does not work when adding an input layer to a trained model. In addition, we found that the application of the downgraded version of Deep SHAP is difficult to apply in the real environment because it takes too much processing time to explain complex deep learning models. Gradient SHAP can operate on multiple input models and can account for their impact on the output results for each middle layer of the model. Thus, it was decided to use Gradient SHAP as an explanation method.

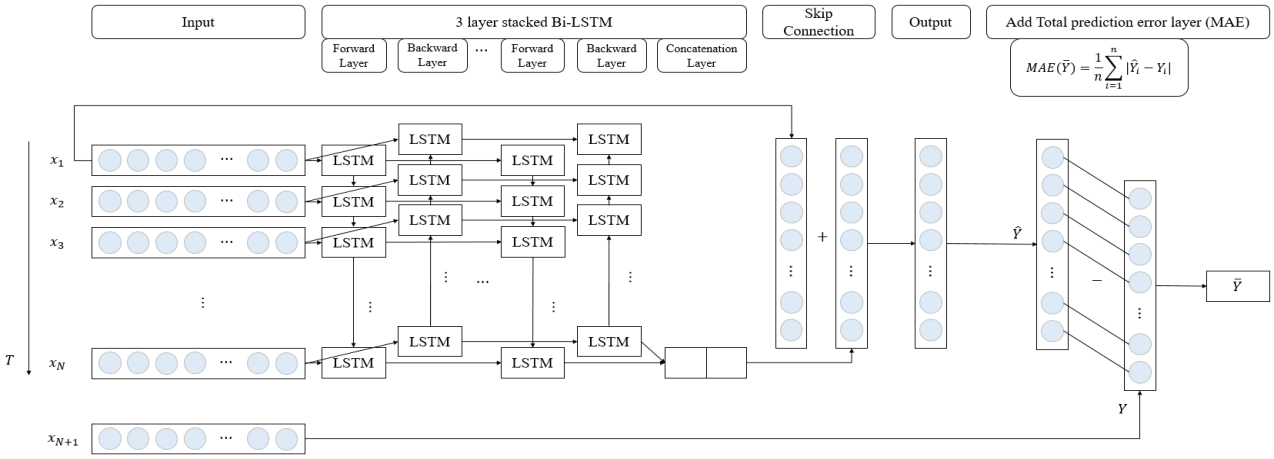


**FIGURE 8.** Adding a layer of the total prediction error to provide explanations for the total prediction error.

In general, in order to interpret the results of the model, the feature with the maximum error between the actual value and the predicted value is considered as the feature that contributed to the prediction. In addition, this study identifies features that contributed to prediction by applying gradient SHAP to a 60-window size model that uses all the features with the best single performance among the anomaly detection models proposed in Section V. The model used is a better model than the first place winner in HAICon2020. The background dataset contains all the training data. All SHAP frameworks compute SHAP values equal to the input dimensions per output node of the model, e.g., if the model has ten output nodes with 2D data (time step, features) as input, the SHAP value of 3D (output nodes, time step, features) is calculated. According to the SHAP value, the features that contributed to the prediction value of each output node can be interpreted. A positive SHAP value indicates that the larger the feature value, the larger the predicted value. Conversely, a negative SHAP value indicates that the larger the feature value, the smaller the predicted value. Our anomaly explanation method is interpreted as the SHAP value of the feature with the highest prediction error. This may explain the anomaly for features with the highest prediction errors. Another way to explain the anomaly is to add an output layer so that each neuron computes the mean prediction error for one feature, as shown in Figure 8. This approach can determine the most important features that affect the total prediction error without retraining the model.

## VI. EXPERIMENT RESULT

### A. SETUP

This study was trained and tested on an Intel Xeon Gold 6226 2.7G server (128 GB of RAM) using the NVIDIA 16GB Tesla T4 GPU. The development environment used the Python 3 programming language in Anaconda 3 Jupyter Notebook. The model and anomaly detection algorithm were implemented using Google's TensorFlow and Keras framework. We used low-level APIs to facilitate fine-grained control of the network architecture.

### B. ANOMALY DETECTION RESULT

We normalized the test data by applying a max-min scaling technique based on the training data. The loss function uses Mean Squared Error (MSE) and the optimization function uses an Adam optimizer. The batch size is 512 and the epoch is 32. The training model is saved only when the loss value decreases during training. Training is terminated if the loss value does not decrease for four epochs. We used the skip connection technique for prediction by summing the first value of the input window and the output of the Bi-LSTM model. The anomaly score is calculated as the MAE of the predicted and actual value. The MA parameter $n$ was set to 10. If there is an ambiguous gray zone because of the threshold, it is predicted with the previous label. In this case, the gray zone smoothing parameter $n$ was set to 10.

#### 1) EVALUATION MOTHED (eTaPR)

Precision, recall, and f1-score are widely used evaluation methods in anomaly detection. However, the accuracy and recall are not sufficient to evaluate whether various types of anomalies have been detected. For example, detecting four anomalies types for 1 second each is equivalent to detecting one anomaly type for 4 seconds. In fact, it is more important to detect many types of anomalies than to accurately detect one type of anomaly.

When using the HAI dataset, it is recommended to evaluate the performance with enhanced Time-series Aware Precision and Recall (eTaPR). The goal of eTaPR is to evaluate the predicted attack range from normal, and it consists of TaP and TaR. TaP is a score on whether the prediction results detect anomalies without false detections, and TaR is a score on how different attack ranges are detected. It also tells you how many attacks were detected in total (should be detected within at least 10% of the attack range).

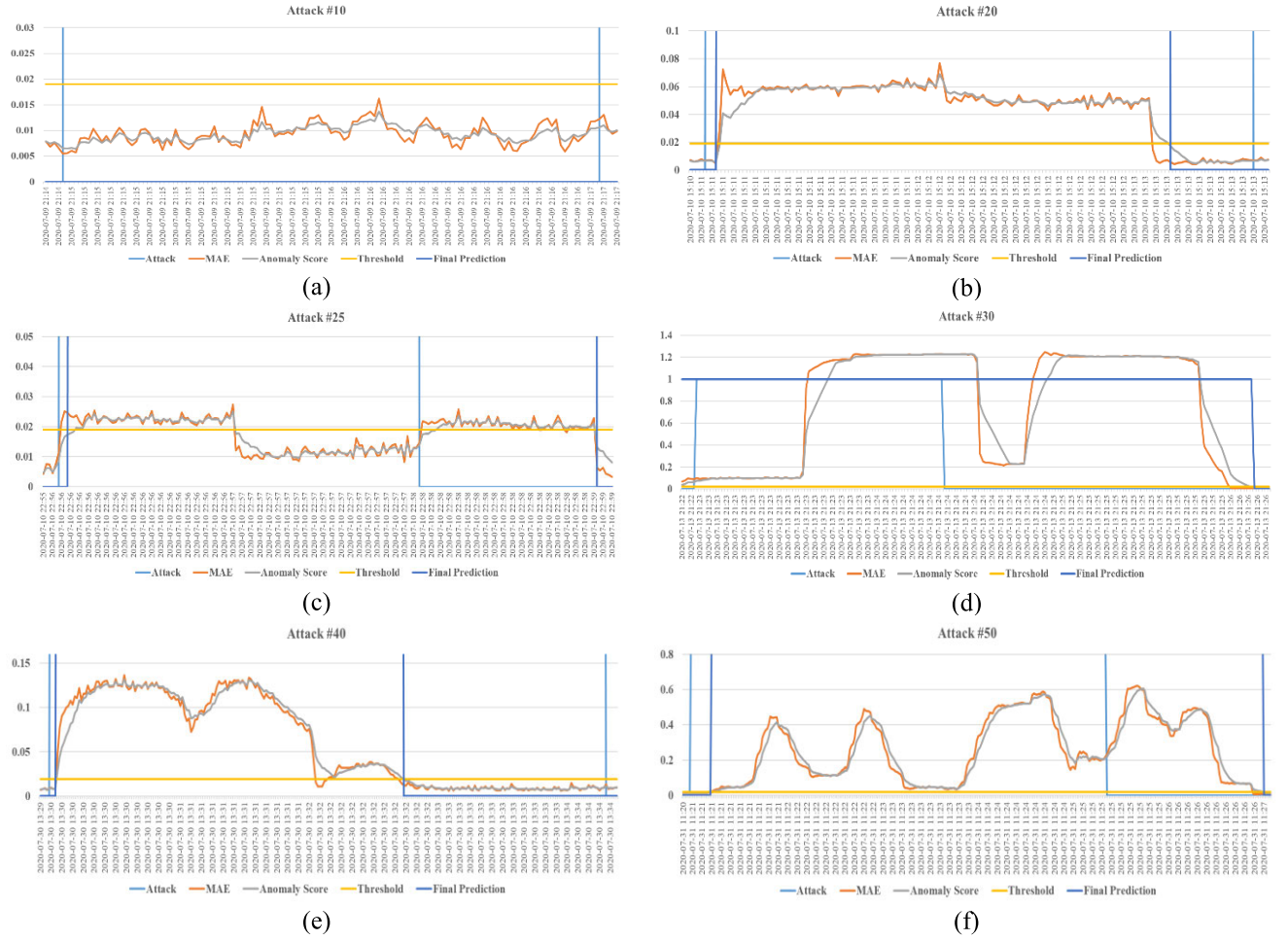


**FIGURE 9.** Anomaly score graph for each Attack type. (a) Attack #10 is not detected; (b) Attack #20; (c) Attack #25; (d) Attack #30; (e) Attack #40; (f) Attack #50.

### 2) EVALUATION RESULTS FOR EACH ANOMALY DETECTION MODEL

Our anomaly detection model has five models with different time steps and four models using only each process feature. Each anomaly detection model gets a threshold through grid search based on the highest F1 score of eTaPR. If the anomaly score is higher than the threshold, it is predicted as an anomaly.

The performance of the five models at each time step is presented in Table 5. The F1 score is lowest when the time step is 55 seconds, and highest when the time step is 59 seconds. It also failed to detect 5 out of 50 anomalies when the time step was 55 seconds, and failed to detect 1 when it was 59 seconds. Therefore, the optimal time step can be set. The performance of the model for each process set to 59 seconds is listed in Table 6. In the HAI dataset, there are abnormal labels for P1, P2, and P3, but there is no label for P4, so it is difficult to accurately evaluate P4 using the entire label. Excluding the results for P4, P3 had the highest F1 score and P1 had the least false detection, but failed to detect various types of abnormalities.

### 3) EVALUATION RESULTS FOR FINAL ANOMALY DETECTION

Our anomaly detection goal is to detect various anomalies by adjusting the threshold to minimize false detections in each anomaly detection model and combining the anomalies detected in multiple models. The combinatorial method works such that if even one model predicts an anomaly, it is considered the final anomaly. First, the 58 time-step model result is added to the 59 time-step model result. Then, the threshold of the 58 time-step model is increased by 0.007 to minimize false detection. In this way, the results of the remaining models are added and the process is repeated. Finally, to minimize false detections, gray zone smoothing is applied to the normal-abnormal sequence predicted in 10 seconds or less, and a final prediction for the anomaly is obtained. The final evaluation result of anomaly detection is presented in Table 7. As the model results were added, TaP decreased little by little to keep false detections to a minimum, and as TaR increased further, it was confirmed that various types of anomalies were accurately detected.

The final anomaly detection result failed to detect one out of 50 attack scenarios (attack #10). The attack detection

**TABLE 5.** Performance of each time-step model.

| Time Step | Threshold | eTaPR | | | |
|---|---|---|---|---|---|
| | | F1 | TaP | TaR | Detect |
| 55 | 0.028 | 0.875 | 0.933 | 0.823 | 45/50 |
| 56 | 0.02 | 0.930 | 0.968 | 0.896 | 48/50 |
| 57 | 0.018 | 0.934 | 0.960 | 0.909 | 48/50 |
| 58 | 0.017 | 0.930 | 0.936 | 0.924 | 49/50 |
| 59 | 0.019 | 0.938 | 0.961 | 0.917 | 49/50 |

**TABLE 6.** Performance of each process model.

| Process | Threshold | eTaPR | | |
|---|---|---|---|---|
| | | F1 | TaP | TaR |
| Bolier(P1) | 0.016 | 0.849 | 0.968 | 0.756 |
| Turbine(P2) | 0.046 | 0.900 | 0.909 | 0.892 |
| Water-Treatment(P3) | 0.019 | 0.939 | 0.934 | 0.945 |
| HIL(P4) | 0.011 | 0.335 | 0.354 | 0.318 |

**TABLE 7.** Evaluation results of final anomaly detection.

| Action | Threshold | eTaPR | | | |
|---|---|---|---|---|---|
| | | F1 | TaP | TaR | Detect |
| 59 Time-Step | 0.019 | 0.938 | 0.961 | 0.917 | 49/50 |
| 58 Time-Step | 0.024 | 0.940 | 0.963 | 0.917 | 49/50 |
| 57 Time-Step | 0.018 | 0.942 | 0.962 | 0.923 | 49/50 |
| 56 Time-Step | 0.02 | 0.943 | 0.960 | 0.926 | 49/50 |
| 55 Time-Step | 0.04 | 0.943 | 0.960 | 0.926 | 49/50 |
| Bolier(P1) | 0.02 | 0.943 | 0.960 | 0.927 | 49/50 |
| Turbine(P2) | 0.051 | 0.943 | 0.960 | 0.927 | 49/50 |
| Water-treatment(P3) | 0.024 | 0.947 | 0.958 | 0.936 | 49/50 |
| HIL(P4) | 0.041 | 0.947 | 0.958 | 0.936 | 49/50 |
| Gray Zone smoothing | - | 0.959 | 0.984 | 0.936 | 49/50 |

criteria for eTaPR evaluation should predict at least 10% of the actual attack range. Figure 9 shows the anomaly scores and final predictions for six out of 50 attack scenarios, including attack #10 in a 59 time-step model. As per the analysis, most attack scenarios are predicted to be more abnormal than the actual attack end point owing to the time-step, indicating that false detections after the actual attack have a low contribution from the actual operation point of view. Figure 9(a) shows a low anomaly score, which was not detected in all models. In this attack scenario, the target feature values exist within the boundaries of the training data, and the actual sensor value is changed, but it is difficult to predict that the change is hidden from the HMI using an AI model. Our final evaluation result is higher than the F1 score of 0.936, which ranked first in HAICon2020 last year. Thus, this approach effectively reduces false alarms by combining positively anomaly predictions (i.e., high anomaly scores) from each model, and allows multiple models to detect a variety of attacks.

### C. EXPLAINABLE SENSOR FAULT DETECTION RESULT

Despite the good performance of anomaly detection results, it is difficult to apply in a real environment because a predicted anomaly does not necessarily mean an actual cyber-attack. Therefore, it is essential to analyze the causes of the results predicted by the AI model, which is time-consuming. The main goal of this study is to support practical quick action and operation by visually indicating the sensors that behave abnormally through XAI-based analysis when anomalies are detected. First, a complex model with some guaranteed anomaly detection accuracy must be analyzed (post-hoc rather than intrinsic). Second, it is necessary to identify the contributing sensors for each prediction time point (Local rather than Global). Finally, it should be applied universally regardless of model (model-agnostic rather than model-specific). As mentioned in section V. C, we use three methods to present explainable results for predicted anomalies. This approach is expected to increase the utility of anomaly detection technology that inevitably accompanies false detection by checking various sensors and actuators suspected in ICS.

Examples of experimental results present the sensors that contributed to the detected and undetected attack sequences. Attack #10 is a short-term (ST) attack that reduces the SP value of P1-PC (Pressure Control) for a few seconds and restores it to normal, and it repeats several times while hiding the SP changes in HMI. The target controller is the pressure control of P1, the target point is P1_B2016. P1-PC pressure controller is a feedback controller for two pressure-control valves (PCV01D and PCV02D) and maintains the pressure (PIT01) between the main and return water tanks according to an operator's SP command (B2016). Attack #20 increases the turbine control value of P2-SC and restores it to normal. The target controller is the speed control process of P2, and the target points are P2_SCO and P2_SIT01. P2-SC speed controller increases the motor speed from zero to the minimum controlling speed at a constant rate, and facilitates engagement control with a proportional integral derivative (PID) controller to maintain a motor speed (SIT01) as close as possible to the speed SP (AutoSD).

#### 1) INTERPRET AS TOP ERROR FEATURE

This method does not use XAI techniques and is the simplest way to explain the anomalies. It defines the features that contributed to the prediction as the top-k features with the error value between the predicted value and the actual value. The features are listed in descending order of prediction error for each feature and the top k among these are identified. If the prediction error for each feature is high, it can be intuitively explained that it has contributed to the anomaly. A visualization of the features and time contributed to attack #10 is shown in Figure 10. The time range that needs to be interpreted is selected, and the contribution of each feature is normalized on a scale of [0, 1] according to the prediction time. Here, (a) is the actual attack label representing the anomaly, (b) is the time contributed in the time range to be interpreted, and (c) is the contributed features in the time range to be interpreted. The color gradation is a feature with a higher contribution as the color is red. This increases and decreases the prediction error, but is less than the threshold, so it is not

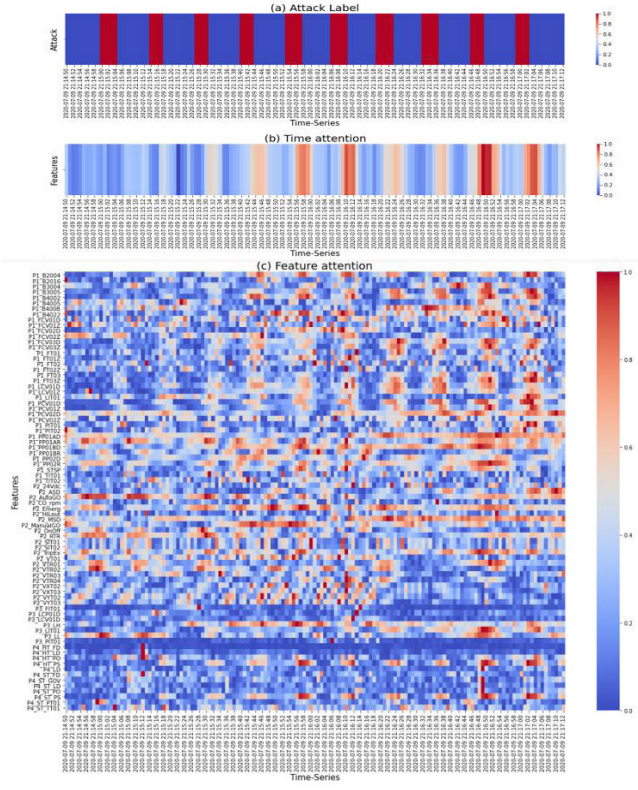


**FIGURE 10.** Contribution of time and sensor based on top error feature for anomaly detection (attack #10).

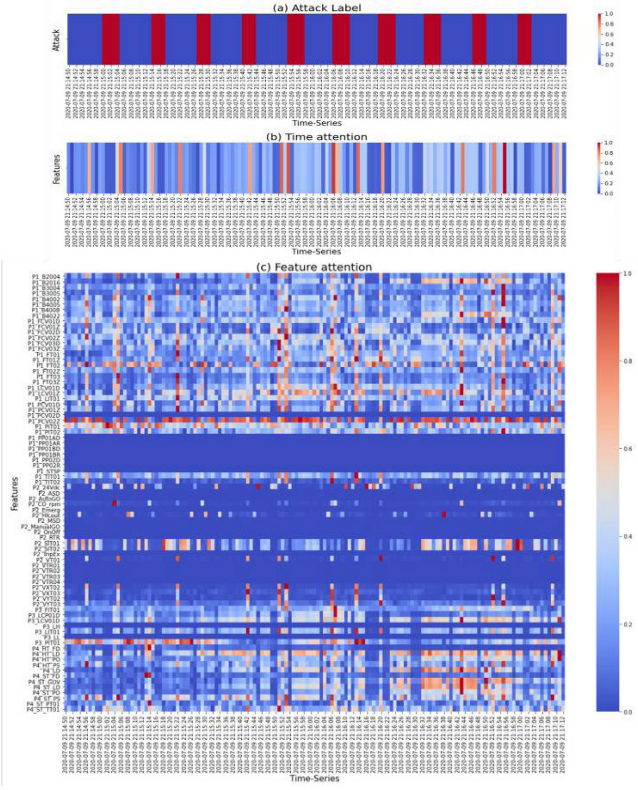


**FIGURE 12.** Contribution of time and sensor based on highest prediction error feature for anomaly detection (attack #10).

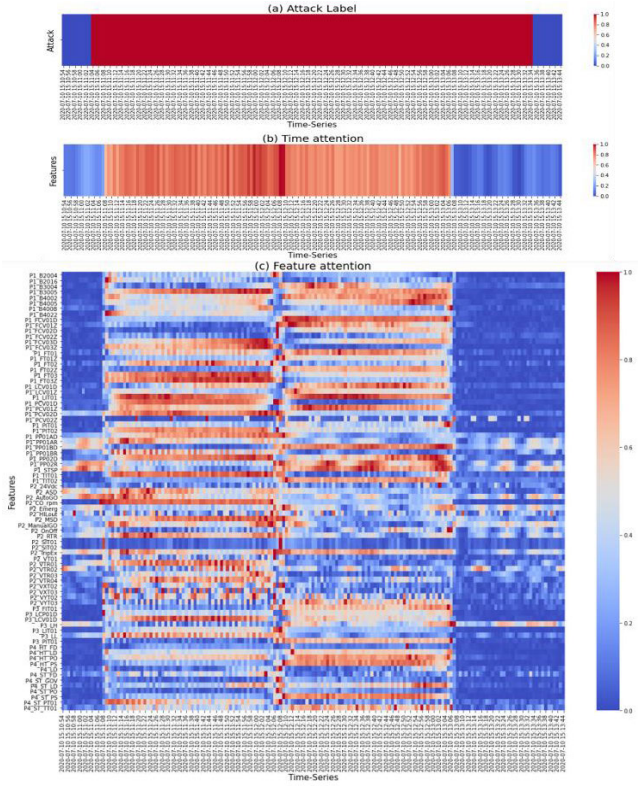


**FIGURE 11.** Contribution of time and sensor based on top error feature for anomaly detection (attack #20).

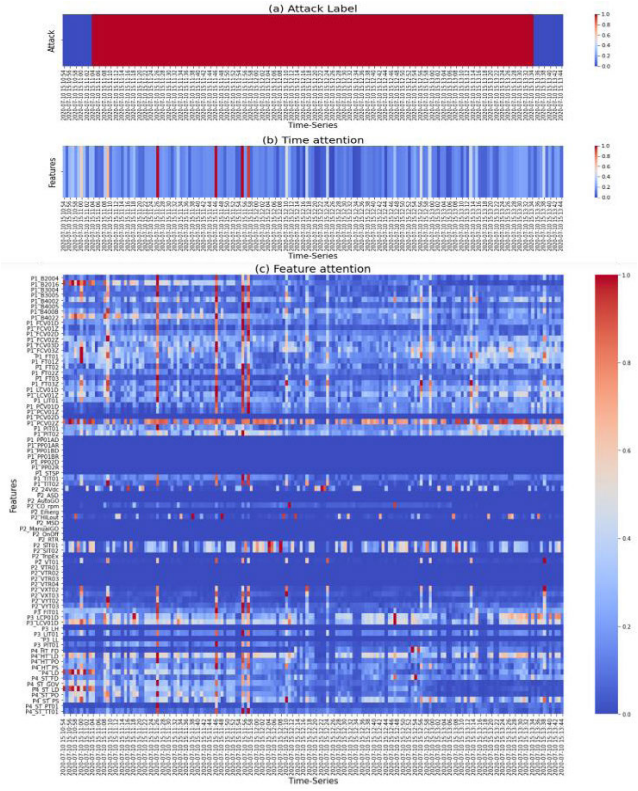


**FIGURE 13.** Contribution of time and sensor based on highest prediction error feature for anomaly detection (attack #20).

predicted as an attack. The contribution of attack #20, which was equally successfully predicted, is shown in Figure 11. Although this approach is straightforward, the information is not clear about which sensors need to be checked, which may force the administrator to check all sensors. As it contributes to all processes, it is not practical and does not fit the purpose of this study.

### 2) INTERPRET WITH THE HIGHEST PREDICTION ERROR FEATURE

This method uses SHAP, one of the XAI techniques, to identify the features that contributed to the top 1 feature, which is the highest prediction error. Contributing features can be interpreted as SHAP values instead of prediction errors. A positive SHAP value indicates that the higher the feature value, the higher the predicted value. Conversely, a negative SHAP value indicates that the higher the feature value, the lower the predicted value. Therefore, it is normalized by taking the absolute value of the SHAP value as it contributes further away from 0. A visualization of the features and time contributing to attack #10 is shown in Figure 12. The contributions towards attack #20, which was equally successfully predicted, are shown in Figure 13. Although this approach can identify unnecessary sensor information through XAI technology, it is difficult to interpret and relies on top 1 features. If the top 1 feature is a false detection, this method is not accurate.

### 3) INTERPRET WITH THE TOTAL PREDICTION ERROR FEATURE

Another method is to add an output layer in such a way that each neuron represents the mean prediction error of one feature. This method identifies the features that contributed to the anomaly score and is faster than the highest prediction error approach, which obtains the SHAP values of all outputs. A visualization of the features and time contributing towards attack #10 is shown in Figure 14. Rather than the top error approach, which contributes to all processes, XAI technology can identify unnecessary sensor information, and in (c), the contribution is distributed upwards, so it can be seen that this is an attack on P1. In addition, over time, the features of P1 show repeated contributions, and the actual attack #10 consists of repeated attacks. In attack #20, as shown in Figure 15 (b), the time taken to restore to normal is clearly indicated as time passes after the attack occurs.

SHAP can perform global analysis by grouping some local analysis. A global summary plot for the specified time range (attack #10 and attack #20) is shown in Figure 16. The summary plot combines feature importance with feature effects. Each point on the summary plot is a Shapley value for a feature and an instance. The position on the y-axis is determined by the feature and on the x-axis by the Shapley value. The color gradation represents the value of the feature from low to high. Overlapping points are jittered in the y-axis direction,

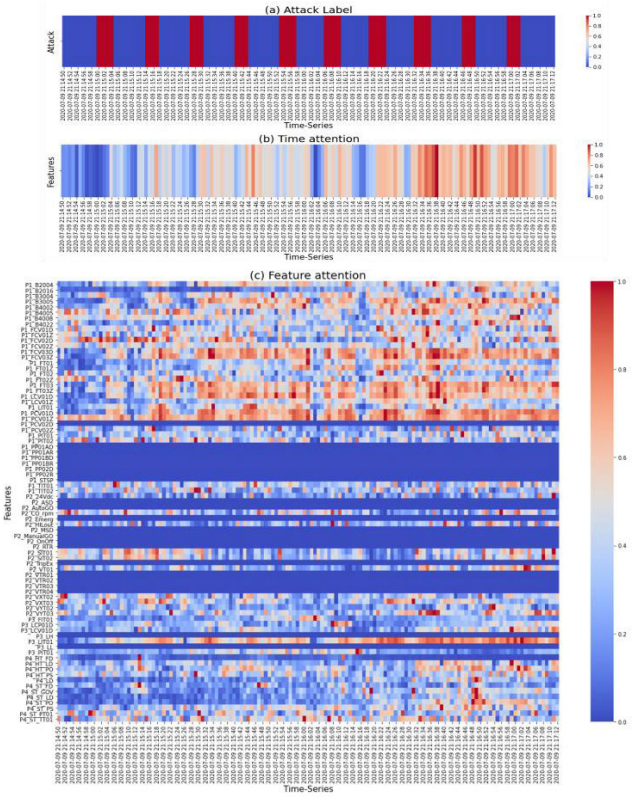


**FIGURE 14.** Contribution of time and sensor based on total prediction error feature for anomaly detection (attack #10).

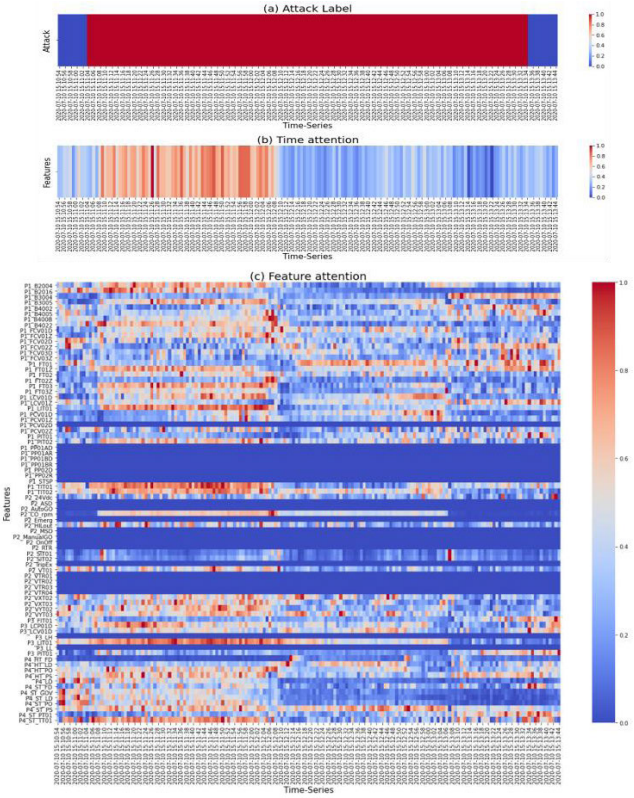


**FIGURE 15.** Contribution of time and sensor based on total prediction error feature for anomaly detection (attack #20).

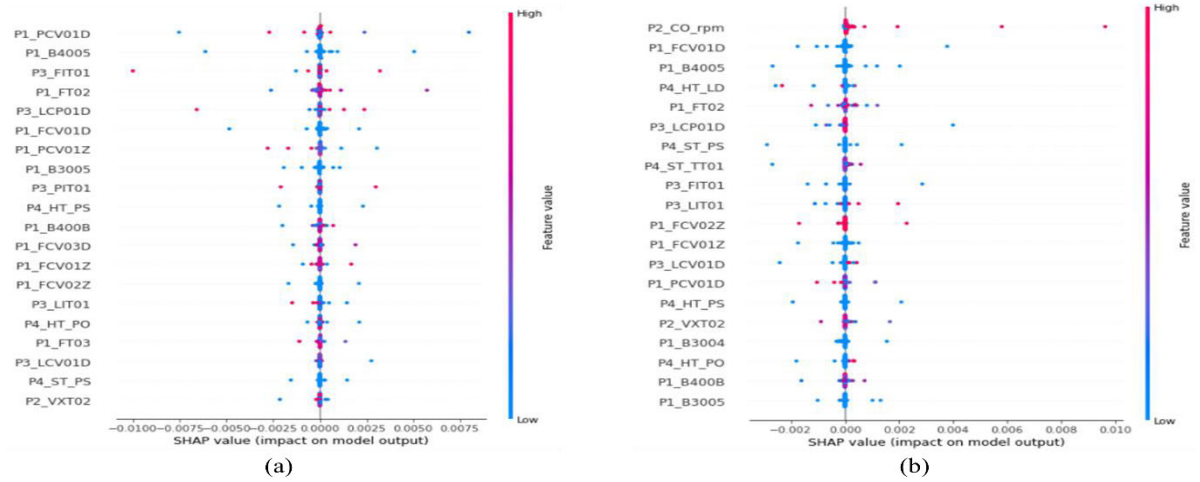


**FIGURE 16.** Summary plot for total prediction error feature. The left (a) is a summary plot for attack #10, and the right (b) is a summary plot for attack #20.

so we get a sense of the distribution of the Shapley values per feature. The features are ordered as per their importance. In (a), lower P1_PCV01D and P1_PCV01Z feature values contribute to anomaly scores predicting attack #10. In (b), overwhelmingly, the higher the P2_CO_rpm feature value, the higher the contribution to attack #20 prediction.

The results of local analysis of the three approaches for explainable anomalies are presented in Table 8. It presents information from the top three explainable sensors for each approach. First, the top error approach for attack #10 has a higher priority of P2 process-related features than P1-PC-related PV and CV features. In attack #20, which is a detected attack unlike attack #10, the first sensor to check is P2_CO_rpm, which indicates the turbine rotations per minute (RPM). Then, as the RPM of the turbine increases, the steam temperature in P4 also rises, so the P4_ST_TT01 sensor

**TABLE 8.** Examples of contributed feature results using three methods for each prediction time.

| Attack Type | Time | Top Error | | | Highest Prediction Error | | | Total Prediction Error | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | Top 1st (Impact) | Top 2nd (Impact) | Top 3rd (Impact) | Highest 1st (Impact) | Highest 2rd (Impact) | Highest 3rd (Impact) | Total 1st (Impact) | Total 2nd (Impact) | Total 3rd (Impact) |
| Attack #10 (2020-07-09) | 21:15:53 | P2_VT01 (0.13412) | P2_VXT02 (0.11096) | P2_VYT03 (0.06352) | P3_LCP01D (0.13216) | P4_HT_LD (0.12473) | P4_LD (0.11695) | P1_PCV01D (0.02322) | P1_B4005 (0.01954) | P1_FCV01D (0.01263) |
| | 21:15:54 | P2_24Vdc (0.12953) | P2_VT01 (0.10241) | P2_VYT02 (0.07119) | P3_LCP01D (0.13220) | P2_HILout (0.10781) | P1_FT02 (0.09029) | P1_PCV01D (0.02154) | P3_FIT01 (0.01059) | P1_B4005 (0.01003) |
| | 21:15:55 | P2_VT01 (0.16155) | P2_VYT02 (0.11890) | P1_PCV01D (0.07845) | P1_FT02 (0.10086) | P2_24Vdc (0.08504) | P1_FCV01D (0.06163) | P1_PCV01D (0.02090) | P1_B4005 (0.01761) | P3_PIT01 (0.01193) |
| | 21:15:56 | P2_VT01 (0.11558) | P2_VYT02 (0.09253) | P1_PCV01D (0.08616) | P3_LCP01D (0.13355) | P4_HT_LD (0.11623) | P4_LD (0.11140) | P1_PCV01D (0.02732) | P1_B4005 (0.01464) | P3_FIT01 (0.01222) |
| | 21:15:57 | P2_24Vdc (0.12918) | P1_PCV01D (0.08974) | P2_CO_rpm (0.06860) | P3_LCP01D (0.12783) | P4_LD (0.11441) | P4_HT_LD (0.11079) | P1_PCV01D (0.02632) | P1_PCV01Z (0.010946) | P3_LCP01D (0.01028) |
| Attack #20 (2020-07-10) | 15:12:18 | P2_CO_rpm (0.562633) | P2_VYT03 (0.17035) | P4_ST_TT0 (0.16356) | P1_PCV02Z (0.07450) | P3_LCP01D (0.06884) | P2_CO_rpm (0.06637) | P2_CO_rpm (0.02171) | P1_FCV01D (0.01155) | P1_B4005 (0.01014) |
| | 15:12:19 | P2_CO_rpm (0.50543) | P4_ST_TT01 (0.19808) | P2_VYT03 (0.18639) | P3_LCP01D (0.07873) | P2_CO_rpm (0.07616) | P1_FT02 (0.07157) | P2_CO_rpm (0.02210) | P1_FCV01D (0.00979) | P1_B4005 (0.00916) |
| | 15:12:20 | P2_CO_rpm (0.55448) | P4_ST_TT01 (0.19575) | P2_VYT03 (0.17933) | P2_VT01 (0.18983) | P2_VXT02 (0.16767) | P1_FCV01D (0.09556) | P2_CO_rpm (0.02327) | P1_FCV01D (0.01343) | P1_FT02 (0.01321) |
| | 15:12:21 | P2_CO_rpm (0.55990) | P4_ST_TT01 (0.18887) | P2_VYT03 (0.16316) | P2_CO_rpm (1.32764) | P3_LCP01D (0.16306) | P1_FT02 (0.12205) | P2_CO_rpm (0.02140) | P1_FT02 (0.01667) | P1_FCV01D (0.01520) |
| | 15:12:22 | P2_CO_rpm (0.49344) | P2_VYT03 (0.15403) | P2_24Vdc (0.12727) | P4_HT_LD (0.07802) | P3_LCP01D (0.07595) | P1_PCV02Z (0.07328) | P2_CO_rpm (0.01895) | P1_FCV01D (0.01622) | P1_FT02 (0.01452) |

should be checked. Second, the approach with the highest prediction error relies on the top 1 features. For example, in attack #10, the topmost features are related to the P2 process, so the electrical-related features of P2 and P4 often appear, but so do other features. In addition, attack #20 explains the top 1 feature, P2_CO_rpm. By changing the CV value for P2-SC, the RPM of the turbine increases and the steam temperature increases. It then gains energy and controls the flow of water to lower the steam temperature. As a result, attack-related features are identified, but not accurate if the topmost feature is a false detection. Finally, the Total Prediction Error approach is most similar in local analysis results in the same attack. Therefore, of the three methods, the interpretation of the total prediction error function is the clearest. Attack #10 is the first thing the pressure control valve PCV01D should check, and it shows clear results. The temperature control feature according to P1-PC and the pressure feature between the water tanks are also given priority. These findings support our decision to use SHAP for explanations. Based on this positive finding, we continued with our method because it provides a more comprehensive explanation to the experts by focusing on the connection between the features with high prediction error and the features that are most important in affecting the mean prediction error. Therefore, it is expected that stable operation of ICS, where availability is important, is possible by checking the sensors that caused the anomaly detected in complex deep learning anomaly detection methods.

## VII. CONCLUSION AND FUTURE WORK

Industrial control systems have migrated from independent network infrastructures and have adopted IT and OT technologies that are accessible through the Internet. Although the efficiency, speed, and precision have increased, this has exposed ICS to the unsecured Internet, rendering the infrastructure open to cybersecurity attacks. To detect anomalies in ICS, multiple Bi-LSTM models for each window size and each process were used and comprehensively analyzed to minimize false detection. The proposed model had an F1 score of 0.959, which failed to detect only one out of 50 attack scenarios, which is higher than the F1 score of the winning HAICon2020 model last year. In the field of security as well as industrial control systems, AI-based anomaly detection research has saved time and money, but it still takes a significant amount of time because direct analysis by security experts is essential. This study provides a comprehensive explanation to security experts by applying SHAP, an XAI technique, to an anomaly detection model and presents an anomaly interpretation without model retraining. Using this explanation, security experts can identify the likely sensors that caused the abnormality during the response process after detecting an anomaly, and by checking these, a faster response and recovery is possible. In the future, we plan to conduct additional studies for sensor fault detection and evaluation in ICS using various XAI techniques.

## REFERENCES


[1] J. Sakhnini, H. Karimipour, A. Dehghantanha, R. M. Parizi, and G. Srivastava, "Security aspects of Internet of Things aided smart grids: A bibliometric survey," *Internet Things*, vol. 14, pp. 1–15, Sep. 2019.

[2] H. HaddadPajouh, A. Dehghantanha, R. M. Parizi, M. Aledhari, and H. Karimipour, "A survey on Internet of Things security: Requirements, challenges, and solutions," *Internet Things*, vol. 14, pp. 1–19, Nov. 2019.

[3] Wired. (2016). *Inside the Cunning, Unprecedented Hack of Ukraine's Power Grid*. [Online]. Available: https://www.wired.com/2016/03/inside-cunning-unprecedented-hack-ukraines-power-grid/

[4] A. Abdallah, M. A. Maarof, and A. Zainal, "Fraud detection system: A survey," *J. Netw. Comput. Appl.*, vol. 68, pp. 90–113, Jun. 2016.

[5] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Jul. 2019.

[6] J. E. van Engelen and H. H. Hoos, "A survey on semi-supervised learning," *Mach. Learn.*, vol. 109, no. 2, pp. 373–440, Feb. 2020, doi: 10.1007/s10994-019-05855-6.

[7] L. Ruff, R. A. Vandermeulen, N. Görnitz, A. Binder, E. Müller, K.-R. Müller, and M. Kloft, "Deep semi-supervised anomaly detection," 2019, *arXiv:1906.02694*. [Online]. Available: https://arxiv.org/abs/1906.02694

[8] L. Maglaras, H. Janicke, J. Jiang, and A. Crampton, "Novel intrusion detection mechanism with low overhead for SCADA systems," in *Security Solutions and Applied Cryptography in Smart Grid Communications*. Hershey, PA, USA: IGI Global, 2016, pp. 160–178.

[9] Y. Zhang, L. Wang, W. Sun, R. C. Green, and M. Alam, "Artificial immune system based intrusion detection in a distributed hierarchical network architecture of smart grid," in *Proc. IEEE Power Energy Soc. Gen. Meeting*, Detroit, MI, USA, Jul. 2011, pp. 1–8.

[10] Y. Zhang, L. Wang, W. Sun, R. C. Green, and M. Alam, "Distributed intrusion detection system in a multi-layer network architecture of smart grids," *IEEE Trans. Smart Grid*, vol. 2, no. 4, pp. 796–808, Dec. 2011.

[11] H.-K. Shin, W. Lee, J.-H. Yun, and H. C. Kim, "HAI 1.0: HIL-based augmented ICS security dataset," in *Proc. 13th USENIX Workshop Cyber Secur. Experimentation Test (CSET)*, 2020, pp. 1–5.

[12] H.-K. Shin, W. Lee, J.-H. Yun, and H. C. Kim. (2021). *HIL-based Augmented ICS (HAI) Security Dataset*. [Online]. Available: https://github.com/icsdataset/hai

[13] K. Stouffer, V. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, "Guide to industrial control system security," U.S. Dept. Commerce, Nat. Inst. Standards Technol. (NIST), Gaithersburg, MD, USA, Tech. Rep. NIST Special Publication 800-82 Revision 2, May 2015. [Online]. Available: http://dx.doi.org/10.6028/NIST.SP.800-82r2

[14] H.-K. Shin, W. Lee, J.-H. Yun, and B.-G. Min, "Two ICS security datasets and anomaly detection contest on the HIL-based augmented ICS testbed," in *Proc. Cyber Secur. Experimentation Test Workshop*, Santa Clara, CA, USA, Aug. 2021, pp. 36–40.

[15] W.-S. Hwang, J.-H. Yun, J. Kim, and H. C. Kim, "Time-series aware precision and recall for anomaly detection: Considering variety of detection result and addressing ambiguous labeling," in *Proc. 28th ACM Int. Conf. Inf. Knowl. Manage.*, Beijing, China, Nov. 2019, pp. 2241–2244.

[16] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI," *Inf. Fusion*, vol. 58, pp. 82–115, Jun. 2020.

[17] A. Das and P. Rad, "Opportunities and challenges in explainable artificial intelligence (XAI): A survey," 2020, *arXiv:2006.11371*. [Online]. Available: http://arxiv.org/abs/2006.11371

[18] D. Gunning and D. Aha, "DARPA's explainable artificial intelligence (XAI) program," *AI Mag.*, vol. 40, no. 2, pp. 44–58, Jun. 2019.

[19] M. Haselmann, D. P. Gruber, and P. Tabatabai, "Anomaly detection using deep learning based image completion," in *Proc. 17th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, Dec. 2018, pp. 1237–1242, doi: 10.1109/ICMLA.2018.00201.

[20] S. Kim, C. Hwang, and T. Lee, "Anomaly based unknown intrusion detection in endpoint environments," *Electronics*, vol. 9, no. 6, p. 1022, Jun. 2020.

[21] N. An, A. Duff, G. Naik, M. Faloutsos, S. Weber, and S. Mancoridis, "Behavioral anomaly detection of malware on home routers," in *Proc. 12th Int. Conf. Malicious Unwanted Softw. (MALWARE)*, Fajardo, PR, USA, Oct. 2017, pp. 47–54.

[22] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar, "Adversarially learned anomaly detection," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Singapore, Nov. 2018, pp. 727–736.
[23] C. Hwang, D. Kim, and T. Lee, "Semi-supervised based unknown attack detection in EDR environment," *KSII Trans. Internet Inf. Syst.*, vol. 14, no. 12, pp. 1–18, Dec. 2020.
[24] M. Kravchik and A. Shabtai, "Detecting cyber attacks in industrial control systems using convolutional neural networks," in *Proc. Workshop Cyber-Phys. Syst. Secur. Privacy (CPS-SPCCCS)*, New York, NY, USA, Oct. 2018, pp. 72–83.
[25] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954–21961, 2017.
[26] C. Wang, B. Wang, H. Liu, and H. Qu, "Anomaly detection for industrial control system based on autoencoder neural network," *Wireless Commun. Mobile Comput.*, vol. 2020, pp. 1–10, Aug. 2020.
[27] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han, "Enhanced network anomaly detection based on deep neural networks," *IEEE Access*, vol. 6, pp. 48231–48246, 2018.
[28] U. Schlegel, H. Arnout, M. El-Assady, D. Oelke, and D. A. Keim, "Towards a rigorous evaluation of XAI methods on time series," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. Workshop (ICCVW)*, Seoul, Oct. 2019, pp. 4197–4201.
[29] A. Morichetta, P. Casas, and M. Mellia, "EXPLAIN-IT: Towards explainable AI for unsupervised network traffic analysis," in *Proc. 3rd ACM CoNEXT Workshop Big DAta, Mach. Learn. Artif. Intell. Data Commun. Netw.*, New York, NY, USA, Dec. 2019, pp. 22–28.
[30] M. T. Ribeiro, S. Singh, and C. Guestrin, "Why should i trust you?' Explaining predictions any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2016, pp. 1135–1144.
[31] A. Patil, A. Wadekar, T. Gupta, R. Vijan, and F. Kazi, "Explainable LSTM model for anomaly detection in HDFS log file using layerwise relevance propagation," in *Proc. IEEE Bombay Sect. Signature Conf. (IBSSC)*, Mumbai, India, Jul. 2019, pp. 1–6.
[32] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153. [Online]. Available: https://proceedings.mlr.press/v70/shrikumar17a.html
[33] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, Oct. 2017, pp. 618–626.
[34] R. Assaf and A. Schumann, "Explainable deep neural networks for multivariate time series predictions," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Macao, China, Aug. 2019, pp. 6488–6490.
[35] X. Song, H. Kanasugi, and R. Shibasaki, "Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, New York, NY, USA, 2016, pp. 2618–2624.
[36] H. Yu, Z. Wu, S. Wang, Y. Wang, and X. Ma, "Spatiotemporal recurrent convolutional networks for traffic prediction in transportation networks," *Sensors*, vol. 17, no. 7, p. 1501, Jun. 2017.
[37] K. Greff, R. K. Srivastava, J. Koutnìk, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space Odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 10, pp. 2222–2232, Oct. 2017.
[38] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Las Vegas, NV, USA, Jun. 2016, pp. 770–778.
[39] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (MAE) over the root mean square error (RMSE) in assessing average model performance," *Climate Res.*, vol. 30, no. 1, pp. 79–82, Dec. 2005.
[40] S. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," 2017, *arXiv:1705.07874*. [Online]. Available: https://arxiv.org/abs/1705.07874
[41] E. Štrumbelj and I. Kononenko, "Explaining prediction models and individual predictions with feature contributions," *Knowl. Inf. Syst.*, vol. 41, no. 3, pp. 647–665, Dec. 2014.
[42] Lundberg. (2019). *SHAP*. [Online]. Available: https://github.com/slundberg/shap
[43] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 3319–3328.

**CHANWOONG HWANG** received the B.S. degree in information security from Hoseo University, Asan, South Korea, in 2020, where he is currently pursuing the M.Sc. degree with the Information Security Department. His research interests include artificial intelligence, intrusion detection, and anomaly detection.

**TAEJIN LEE** received the bachelor's degree from the Pohang University of Science and Technology (POSTECH), in 2003, the master's degree from Yonsei University, in 2008, and the Ph.D. degree from Ajou University, in 2017. He worked at the Korea Internet Security Agency, from 2003 to 2017. He has been working with Hoseo University, since 2017. His research interests include intrusion tolerance technology, VoIP/Wibro security, malware distribution detection/analysis, e-mail security, cyber black box, malware profiling, and mobile payment fraud detection. His current main research interests include artificial intelligence, malicious code analysis, and intrusion detection.