QATAR UNIVERSITY

COLLEGE OF ENGINEERING

MULTIMODAL INTRUSION DETECTION SYSTEM FOR CYBER PHYSICAL SYSTEMS

BY

SOHAILA SALAH ELTANBOULY

A Thesis Submitted to

the College of Engineering

in Partial Fulfillment of the Requirements for the Degree of

Masters of Science in Computing

June 2021

COMMITTEE PAGE

The members of the Committee approve the Thesis of
Sohaila Eltanbouly defended on 19/04/2021.

Dr. Abdelkarim Erradi
Thesis/Dissertation Supervisor

Dr. Erchin Serpedin
Committee Member

Dr. Khaled Khan
Committee Member

Dr. Khaled Shaban
Committee Member

Dr. Nader Meskin
Committee Member

Approved:

Khalid Kamal Naji, Dean, College of Engineering

ABSTRACT

ELTANBOULY, SOHAILA S., Masters : June : [2021],

Masters of Science in Computing

Title: Multimodal Intrusion Detection System for Cyber-Physical Systems

Supervisor of Thesis: Abdelkarim, Erradi.

Cyber-Physical Systems (CPS) are deployed to control critical infrastructure in many fields, including industry and manufacturing. In recent years, CPS have been affected by cyberattacks due to the increased connectivity of these systems to the Internet. This work aims to develop a deep learning-based Intrusion Detection System (IDS) for detecting cyberattacks on CPS using multimodal learning techniques. This thesis reports the design, implementation, and evaluation of two IDS solutions based on different deep learning networks: Convolution Neural Network (CNN) and Recurrent Neural Network (RNN). For the first IDS, Gramian Angular Field (GAF) is used to convert CPS time-series data to images that are fed to a 3D CNN to train the attack detection classifier. The second IDS uses RNN with a multimodal attention approach for training the attack detector. Both solutions utilize CPS process data and network data to improve the attack detection accuracy. The performance of the proposed approaches is evaluated on SWaT datasets collected from a testbed that represents real-world CPS. Experimental results demonstrate that both IDSs achieved improved performance and higher detection capability compared to related work.

DEDICATION

To my parents for their prayers, love, and faith in me.

To my three beloved sisters for their encouragement and support through the difficult times.

ACKNOWLEDGMENTS

In the name of Allah, the Most Gracious, the Most Merciful

All praise and glory to Almighty Allah who granted me this opportunity and gave me the strength and patience to complete this thesis.

I would like to express my deepest gratitude to my supervisor Dr. Abdelkarim Erradi for his constant guidance to carry out this work. This work would not be possible without his supervision and valuable feedbacks. I would like to extend my gratitude to Dr. Ashraf Tantawy for his continuous help, feedback, and ideas, which contributed significantly to the improvement of this thesis. My appreciation is also extended to Dr. Ahmed Ben-Said for his collaboration and valuable advice.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ACRONYMS

CPS	Cyber Physical System
IDS	Intrusion Detection System
CNN	Convolution Neural Network
RNN	Recurrent Neural Network
GRU	Gated Recurrent Unit
GAF	Gramian Angular Field
GASF	Gramian Angular Summation Field
GADF	Gramian Angular Difference Field
MTS	Multivariate Time-Series
ReLU	Rectified Linear Unit
CBP	Compact Bilinear Pooling
HMI	Human Machine Interface
ICS	Industrial Control System

CHAPTER 1: INTRODUCTION

Cyber-Physical Systems (CPS) are composed of interconnected physical and cyber elements to monitor and control critical physical processes in different areas [1]. Some of the main application fields of CPS are healthcare, manufacturing, industry, and transportation. Historically, these systems were designed to operate at physically isolated locations on proprietary hardware and software. CPS encompasses different control systems, including supervisory control and data acquisition (SCADA) systems and programable logic control (PLC). Figure 1 illustrates the basic operations of industrial CPS according to NIST's Industrial Control Systems Security Guide [2]. At the bottom layer of the architecture are the control loops that utilize sensing and actuation components to directly control and manipulate the process. The sensors and actuators are connected to PLCs, where the control algorithms are implemented. The PLCs receive and interpret the sensors' signals. Then, based on the control algorithm and the target set points, they send back the required values to the actuators to adjust the process to the desired state. At the top level, there are the Human Machine Interfaces (HMIs) and the maintenance facilities. The operators use the HMIs to monitor and control the process via the PLCs as well as configuring and maintaining the control algorithm, set points, and all other parameters needed for the process operation.

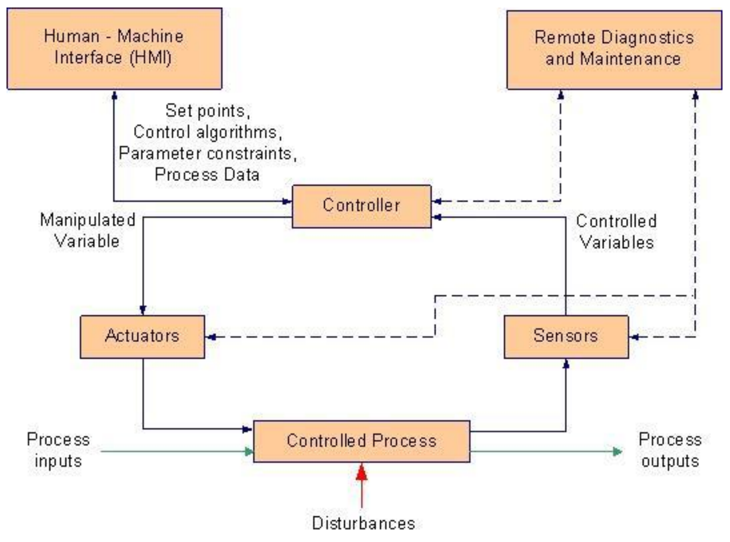
Human - Machine Interface (HMI)
Remote Diagnostics and Maintenance
Set points, Control algorithms, Parameter constraints, Process Data
Controller
Manipulated Variable
Controlled Variables
Actuators
Sensors
Process inputs
Controlled Process
Process outputs
Disturbances

Figure 1: Industrial CPS operation [2].

Recently, with the evolution in information and communications technology (ICT), these systems have been increasingly connected to corporate networks using standard protocols and hardware/software components to reduce cost and to allow remote monitoring and control. However, despite the great functionalities that this connectivity provides, it introduces new challenges in operating these systems and exposes them to vulnerabilities. Also, since the adapted technologies are coming from different vendors, new weaknesses have emerged that may cause the system's failures. Furthermore, many of the industrial communication protocols that have been used for communication between the different devices in the control network lack security mechanisms such as encryption and integrity [3]. Due to these vulnerabilities, intruders can target the system with cyber-attacks to gain unauthorized access to the control network and cause disturbances in the physical process. In the last decade, several sophisticated malware targeted industrial infrastructures, such as the Stuxnet worm that hit Iran's nuclear infrastructure in 2010 and the Shamoon virus that targeted the Saudi Arabia oil company Aramco [4]. Moreover, in Qatar, an unknown virus targeted Ras

Gas company and hit its computer systems forcing the network to be offline for days [5].

Due to the severe consequences that such attacks may cause on the society, environment, and economy, security mechanisms are essential for CPS to ensure the stable operation of the process and avoid the catastrophic impacts of cyberattacks. Fault diagnosis systems, which detect any deviations between the actual and a simulated process, are not sufficient for security protection in CPS and need to be combined with Intrusion Detection Systems (IDS) for effective protection. IDS have been widely investigated for monitoring CPS security and for detecting cyber-attacks. According to the National Infrastructure Security Coordination Centre (NISCC), firewalls and IDS are recommended for SCADA systems and control networks to monitor the system processes and network traffic [6].

IDS provides a protection layer for CPS by monitoring and analyzing the system's events to spot any signs of abnormality. The system's behavior should be continuously monitored to ensure the system's stable operation at the physical and network levels. Modeling the behavior of industrial processes accurately is a very challenging and time-consuming process that requires a deep understanding of the underlying process and its implementation details. Also, the IDS built to model an industrial process usually cannot be generalized for usage with other processes. Recently, to overcome these limitations, several CPS datasets have been created to enable the learning of complex models and behavior-based IDS using data mining techniques. Additionally, deep learning has been widely used for handling high-dimensional data and extracting better feature representations.

Multiple research efforts considered the design of IDS for CPS; however, most of them have focused on attack detection at one layer of the CPS, such as the network layer or the physical layer. Since CPS are multilayer systems, the cyber-attack effect can be seen at one or multiple layers. Thus, deploying an IDS that considers the data from the different layers and different data sources across the CPS would enhance the detection accuracy. In this thesis, two different IDS are designed and developed using deep multimodal learning techniques that combine CPS process data and network data.

1.1. Problem Statement

CPS controls critical industrial infrastructure; therefore, any cyberattacks targeting these systems may cause catastrophic economic and social impacts if no proper protection mechanisms are applied. Thus, IDS is usually deployed to monitor the system's behavior to detect abnormal behaviors. For CPS, model-based IDS solutions are widely considered by modeling the behavior of the system. For example, in [7], the normal behavior of the CPS is modeled using multiple synchronized hybrid automata, which is then used to define IDS rules. However, modeling industrial processes accurately is a complex task that requires a deep understanding of the underlying process and a consideration for all the scenarios that may occur in the system. Due to these systems' complexity and the unknown disturbances during normal operation, the model-based IDS may suffer from low accuracy. On the other hand, signature-based IDS is also widely used by defining a set of patterns or rules for known attacks and violations. For example, a set of intrusion detection rules based on Modbus protocol specification and known attack signatures are defined in [8]. Even though signature-based IDS provide a low false alarm rate, but they come with the cost of incapability of detecting unknown attacks. This thesis focuses on building a multimodal IDS for CPS to detect cyber-attacks and any abnormal behaviors using deep multimodal learning techniques. The multimodal techniques could increase the attacks' discoverability and provide better detection over other techniques by leveraging data from multiple sensors distributed across the CPS. More specifically, this thesis answers the following two research questions:

- Can deep neural networks with multimodal techniques provide better performance for detecting CPS cyberattacks compared to other techniques?
- How can deep multimodal techniques be adopted for building IDS for CPS?

1.2. Thesis Objectives

In this thesis, we aim to design a multimodal IDS that considers data from different CPS layers and can detect cyberattacks with higher accuracy. Multimodal deep learning approaches are investigated to combine data from different sensors to train a better attack detector. These techniques range from the simple concatenation of data at the input layer of the network, the fusion of the decisions from multiple classifiers at the output layer, to the learning of a hierarchical representation of the data across the network's hidden layers [9]. The focus of the thesis is on the later approach, which helps in learning a joint representation and finding the relationships between the different inputs.

The aim of this research can be achieved through the following objectives:

- Develop a CNN-based IDS trained on CPS time-series data preprocessed using Gramian Angular Field imaging. The IDS uses 3D CNN to capture better dependencies between the different time series along with using different merge techniques to fuse the data from different input sources.
- Develop a multimodal IDS based on RNN and multimodal attention approach that considers different input modalities.
- Conduct a comparative study with several state-of-the-art techniques using different evaluation metrics and datasets to validate the effectiveness of the proposed approaches.

1.3. Thesis Contribution

In this thesis, the problem of CPS cyberattacks detection is tackled using two different approaches. In the first approach, named IDS-ITS, we developed and evaluated an IDS based on Imaging Time Series. The approach is based on converting the time series of the sensors' and actuators' readings to images using the Gramian Angular Field (GAF) method, then training a CNN classifier to detect attacks. The CNN-based solution is selected because of its effectiveness in image classification tasks (e.g., [10] and [11]). This approach aims to leverage and incorporate the advanced techniques of computer vision for time series classification by obtaining a visual representation of the time series that captures the temporal correlation between the data points in order to enhance the accuracy of attacks detection.

In the second approach, named IDS-MAN, we developed and evaluated RNN-based IDS that leverages Multimodal Attention Network (MAN). The approach uses the GRU alternative to avoid the vanishing gradient problem in standard RNN [12]. Each input is processed separately by GRU layers, then the Bi-Modal framework [13] is used to leverage the multimodal information from each pair of modalities. The model is trained using supervised learning. However, to provide a better generalization to detect unseen attacks, embedding vectors for the classes are learned, and the detection output is determined by the distance between the classes' embeddings and the input. The motivation of using attention for multimodal learning is to leverage the most significant information from the different data sources for an efficient fusion. Recently, Multimodal Attention mechanisms were successfully applied in a wide range of Natural Language Processing (NLP) tasks such as video description [14], visual question answering [15], and sentiment analysis [13].

The contribution of the thesis can be summarized as follows:

- Design, implement and evaluate an IDS for CPS based on a two-pathways 3D CNN trained on imaging time-series produced using the Gramian Angular Field that converts sensors' and actuators' measurements and network traffic features into images. This representation enables the use of CNN, which is known for its superiority in image classification tasks. The evaluation of the proposed IDS demonstrates its superiority in terms of better attacks' discoverability by detecting some attacks that were not detected previously by related works.
- Examine and evaluate four different multimodal merge techniques to enhance the IDS performance by combining the CNN two-pathways of process-level and network traffic features.
- Design, implement and evaluate a multimodal IDS based on GRU that considers both CPS process data and network data as independent modalities. Then, use a multimodal attention approach to capture the inter-modality relations between each pair of modalities.
- Enhance the generalization of attacks detection and compensate for the lack of labeled data for attack instances by developing an approach to learn embedding vectors for the normal and abnormal classes and perform the anomaly detection based on the cosine similarity.
- Compare the two proposed models with other state-of-the-art techniques and evaluate the proposed models using different datasets.

1.4. Thesis Outline

This chapter presents an overview of the research problem along with the thesis objectives and contributions. The remainder of this thesis is organized as follows:

CHAPTER 2: Provides the background of the IDS techniques and the deep learning approaches along with the related work.

CHAPTER 3: Explains the design and implementation of the proposed IDS-ITS.

CHAPTER 4: Explains the design and implementation of the proposed IDS-MAN.

CHAPTER 5: Presents the evaluation results of the developed IDSs.

CHAPTER 6: Concludes and presents an agenda for future work.

2.1. Background

2.1.1. Intrusion Detection Systems

Intrusion Detection Systems (IDS) provide a protection layer over CPS. Since CPS are multilayer systems, the IDS can be deployed at different areas of the system. The main two sources of data for IDS in CPS are the network traffic and the process data. Moreover, there are two main categories of IDS. The first one is the knowledge-based IDS which looks for events that match specific misbehavior. The second one is the behavior-based IDS that looks for events that deviate from the system's ordinary behavior. Figure 2 shows the different intrusion detection techniques for CPS.

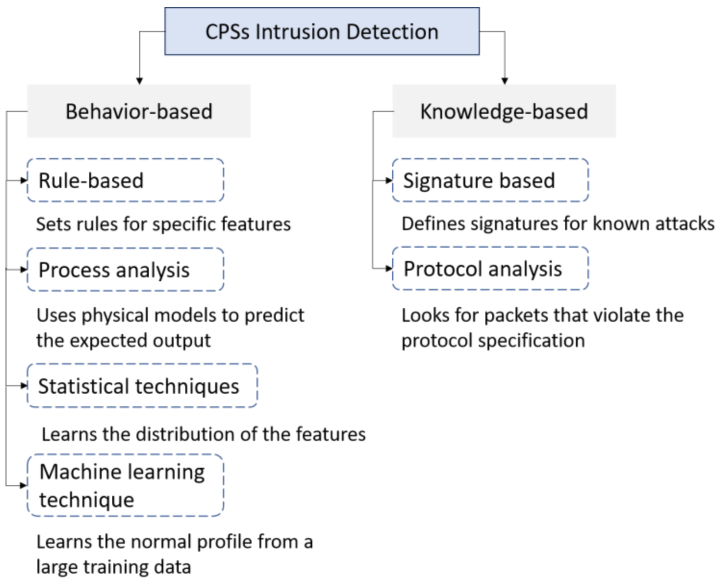
CPSs Intrusion Detection
Behavior-based
Rule-based
Sets rules for specific features
Process analysis
Uses physical models to predict the expected output
Statistical techniques
Learns the distribution of the features
Machine learning technique
Learns the normal profile from a large training data
Knowledge-based
Signature based
Defines signatures for known attacks
Protocol analysis
Looks for packets that violate the protocol specification

Figure 2: Intrusion detection techniques for CPS.

2.1.1.1. Knowledge-Based IDS

This type of IDS needs a predefined and well-established knowledge of specific patterns or misbehavior before implementing the IDS. This type of IDS can be categorized into signature-based and protocol analysis-based. In the signature-based

IDS, the signatures for the known attacks are defined. Then, the IDS compares the observed events' signatures with the predefined patterns to detect anomalies. This method requires a large amount of memory because all the signatures should be saved. Also, it fails in detecting the threats with unknown signatures [16]. The second type of knowledge-based IDS is the protocol analysis-based IDS. Each protocol has a specification that defines its format and communication patterns. This IDS depends on the protocol's specification and looks for packets that violate the protocol's normal activity.

The two types of knowledge-based IDS can be combined. For example, in [8], a set of intrusion detection rules based on Modbus protocol specification and known attack signatures are defined. The IDS raises an alert if the coming traffic matches the predefined signatures or if the rules are violated. To build this IDS, a vulnerability analysis was conducted and a detailed understanding of the protocol requirement and specification was needed to define the rules. However, even though the knowledge-based IDSs have a low false alarm rate, any attack that does not match any defined signatures will be undetected. Thus, recently all the research works have been focusing on the behavior-based IDS or the combination of both techniques.

2.1.1.2. Behavior-Based IDS

Instead of using predefined knowledge, the behavior-based IDS looks for events that deviate from the system's ordinary behavior. There are different behavior-based IDS categories, which are rule-based, statistical techniques, process-analysis, and machine learning and deep learning techniques. The rule-based IDS works first by knowing some prior knowledge, such as the distribution of the data. Then, the detection mechanism relies on a set of fixed rules. However, contrary to the signature-based IDS, each rule is set for one specific metric in the system; for example, hop count and traffic arrival rate can be used as rules to detect many kinds of attacks. The rule-based anomaly detection technique is computationally cheap, simple, and fast. Another type is the statistical-based IDS which is similar to the rule-based IDS in which it learns the underlying distribution of the data. An example of statistical-based IDS is inspecting if a sensor reading is within a specific range. Because of the similarities between the rule-based and statistical-based IDS, some research considered the combination of them. For example, He et al. [17] proposed a rule and statistical-based IDS. First, a set of rules for specific network features is defined, including features at the packet layer, the flow layer, and the inter-flow layer. Then, statistical analysis based on the system's historical behavior is conducted to extract long-term metrics. Then these metrics are used to detect anomalies and trends that do not fit with what is normally observed. All the extracted features in the four layers of the IDS are used to profile the system's normal behavior, and the anomalies are detected if they deviate from this profile.

Another type of behavior-based IDS is the process-analysis based which builds physical models to predict the expected output of the process. Although this type of IDS detects the process's abnormal behavior with very high accuracy, modeling the industrial processes accurately is very challenging because it needs a deep analysis of the underlying process. In [18], the authors study the Linear Parameter Varying (LPV) CPS for the scenarios of false data injection attacks. A system model is obtained based on the LPV state-space model. The model can predict the expected future sensor measurements. Then, the attack detection is performed by obtaining an estimate of the system's state and calculating the difference between the output of the attack detector and the system's output. Finally, the anomalies are detected based on specific thresholds.

The last type of behavior-based IDS is using machine and deep learning techniques. There are three categories for this type which are classification, forecasting, and unsupervised. First, in the **classification**, the IDS categorizes the data into different classes. This type requires a dataset with sufficient anomaly samples. An example of this IDS type is [19], where different machine learning algorithms are compared, namely support vector machine (SVM) random forest, k-nearest neighbor, and k-means clustering for anomaly detection using Modbus protocol datasets. Moreover, in [20], an anomaly detector for the process data based on a probabilistic neural network (PNN) was proposed to map the input variables to output classes.

Second, the **forecasting technique** is based on time-series prediction for the normal behavior of the system. Then the anomalies are detected if the difference between the actual time series readings and the predicted values exceeds a threshold. Many deep learning algorithms have been used for this type of IDS. For example, Long Short Term Memory (LSTM) is used by [21], 1D Convolution Neural Network (CNN) is used in [22], and Multi-Layer Perceptron (MLP) is used in [23].

Third, the **unsupervised technique** creates a representational model for the normal behavior of the system. For example, in [24], two unsupervised machine learning methods are compared for anomaly detection using process data which are LSTM based Deep Neural Network (DNN) and one-class SVM. For the network traffic, Dong and Peng [25] used one-class SVM to build an anomaly detector for Modbus traffic. Moreover, Li et al. [26] proposed a Multivariate Anomaly Detection framework with Generative Adversarial Networks (GAN).

2.1.2. Deep Learning Techniques

2.1.2.1. Convolution Neural Network (CNN)

CNN is a specialized type of Neural Network that is mostly used in the computer vision field to process image data. The core component of the CNN is the convolution layer. The layer has several filters that are smaller in size than the input. Each filter moves across the input from top left to bottom right. The filter performs the convolution operation, which is the element-wise multiplication between the filter's weights and the input values. Then, the results are summed to represent all the learned features for this region. After the filter passes across the input, a features map is generated from each filter. For the output of the layer, all the feature maps resulting from the convolution layer are concatenated. The convolution layer's output is passed to an activation layer (e.g., ReLU, Tanh) to introduce non-linearity in the network. CNN also contains pooling layers that downsample the feature map's size by choosing one number (either the maximum of the average) from each map region to allow the network to train faster and focus on the most important features. Even though CNN is designed for 2-dimensional data, they are extended to be used for 1-dimensional and 3-dimensional data.

2.1.2.2. Gated Recurrent Unit (GRU)

GRU is a type of RNN. The idea behind the RNN is to process sequential data by performing the same computation to every element in the input sequence, taking into consideration the computations of the previous elements. GRU was introduced by Cho, et al. in 2014 [12] to solve the vanishing gradient problem in vanilla RNN. The unit consists of two gates: the reset and update gates. The gates are updated as follow:

$$r_t = \sigma(W^r x_t + U^r h_{t-1})$$

$$z_t = \sigma(W^z x_t + U^z h_{t-1})$$

where W^r, U^r, W^z, and U^z are trainable weights, and σ is the sigmoid function. Both gates have similar equations; however, each has its own set of weights and differs on how its output is used. The variables x_t and h_{t-1} represent the input of the current timestep and the hidden state of the previous unit. The reset gate determines how much to forget from the information of the previous timesteps. The following equation represents this operation:

$$h'_t = \tanh(Wx_t + r_t \odot Uh_{t-1})$$

when r_t is 0 the corresponding information in h_{t-1} will be forgotten, indicating irrelevant information to the future timesteps. The update gate is used to determine which past information needs to be considered for future timesteps and which new information needs to be added. h_t is calculated as follows:

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot h'_t$$

2.2. Related Work

2.2.1. Anomaly-based Intrusion Detection Systems (AIDS)

2.2.1.1. IDS for CPS

Recently, deep learning techniques have been widely deployed for anomaly detection using various neural network types. Long Short Term Memory (LSTM) RNN has been used by Goh et al. [21] for cyber-attacks detection in CPS by modeling the system's normal behavior. Then, the differences between the actual and the ideal sensors values that the RNN predicted are calculated. After that, the cumulative sum method is used to detect any small deviation that could correspond to anomalies. RNN was also used by [27] in a sequence-to-sequence encoding-decoding approach. The time series are encoded to predict the next values, while the decoder is used to predict the future operational data using the attention method. Then, the difference between the prediction and the actual data is calculated to determine the anomalies.

Convolution Neural Network is used in [28] combined with Isolation Forest models to build a hybrid attack detection framework for ICS. The proposed framework consists of two stages trained independently: the features extraction stage using 1D-CNN and the detection stage using the Isolation Forest. An autoencoder based on 1D-CNN is trained to extract features from the continuous-time signals (i.e., sensor measurements). The Isolation Forest model is then trained to perform the detection using the extracted features combined with the actuator's signals. In [22], an anomaly detection approach is proposed based on measuring whether the predicted values deviate from the observed values using 1D CNN. CNN is trained as a time series prediction model to predict the future values of the features. Then, to detect the anomalies, the differences between the predicted and actual values are calculated, and the z-score of each feature is compared to empirically predefined thresholds to determine anomaly existence.

Other types of neural networks are used for IDS in CPS. For example, an anomaly detector for critical infrastructure based on a probabilistic neural network (PNN) was proposed in [20]. The PNN consisted of four layers, and the Bayesian strategy is used to map the input variables to output classes. Moreover, Li et al. [26] proposed a Multivariate Anomaly Detection framework with Generative Adversarial Networks (MAD-GAN). To detect anomalies, both the GAN's discriminator and generator were exploited. The discriminator is used to classify the time series input and the generator is used to measure the residual between the input and the reconstructed samples by the generator. Then, the two losses are combined to detect possible anomalies in the data. Shalyga et al. [23] used a genetic algorithm to find the best neural network architecture for anomaly detection. The best model was based on the multilayer perceptron. The detection method is based on time series forecasting, and the mean error is used to report an anomaly if the error is greater than the 99th percentile of the mean error.

In [24], two unsupervised machine learning methods are compared for anomaly detection. The first method is LSTM based Deep Neural Network (DNN), which is used to implement probabilistic outlier detection. The second method is the one-class Support Vector Machine (SVM) used with the Radial Basis Function kernel to learn non-linear classification boundaries. DNN outperformed SVM with an improvement of 0.4 in terms of F1-score. Lin et al. [29] used a different approach for anomaly detection by building a one-class classifier using a graphical-based approach (TABOR). Timed automata are used to model the behavior and the dynamics of the sensors and actuators. Then, Bayesian Networks are trained to capture the dependencies between the variables. TABOR works as a one-class classifier to detect any deviations from the normal learned behavior.

2.2.1.2. Attention-based IDS

The authors in [30] proposed an anomaly detection model for an ICS network in a real intelligent charging station and power supply system. The proposed model aimed to leverage the sequence relationship between network packets by capturing multi-dimensional features from the traffic using Multi-Head Attention (MHA). The MHA helps in learning information from different traffic packets from different representation subspaces. The scaled dot-product is used as the attention function, and a residual connection is used to add the output of the MHA attention to the original data. The MHA layer's output is fed to a global average pooling and fully connected layers for classification. To evaluate the proposed model, two laboratory-scale State Grid ICS that reflect the industrial control network's characteristics are used. The results showed that the MHA model outperforms other methods such as LeNet, VGG, and AlexNet by an F1-score of 0.998.

An anomaly detection system of multi-stage attention with LSTM-based CNN is proposed in [31] for connected and automated vehicles. First, a CNN is used for the feature extraction, and the produced feature maps are converted into vectors to be fed to the Attention LSTM model. The softmax function is used for the attention weights, with Tanh as the alignment function. The attention mechanism aimed to analyze the data relationship and give scores to the data sequence based on their significance. The dataset used for evaluation included three sensors readings and four types of anomalies with different magnitudes. The results showed superior performance than other models even when the anomalous data had low magnitude in the input stream.

An unsupervised attention-based deep learning framework is proposed in [32] for anomaly detection in network payloads and syscall traces. The proposed framework has three components. The first is the local model, which is implemented using either RNN or AutoEncoder (AE). The model used fixed-length subsequences to extract local features from the original input variable-length sequence. The second component constructs the normality clusters to represent the profile of the normal data. For clustering, the K-means algorithm is used to generate different clusters from the original training data to account for the normal data distribution irregularities. The third component is the attention network which aggregates the local feature vectors based on the normality. The additive attention is used to compute the correlation scores between the local feature vectors and the cluster centers. A global feature vector for each cluster is then calculated by the summation of the local feature vectors weighted by their attention distribution. For anomaly detection, the maximum likelihood estimation is used to select the final global representation from the set of global feature vectors for each cluster, and the vector with minimum distance is selected. The negative likelihood is used as the anomaly score to measure the distance between the sequence and the center of the clusters. The sequence is considered anomalous if the score exceeds a user-defined threshold. The framework is evaluated on the CSIC-2010 and ADFA-LD datasets. The results showed superior performance compared to previous work and non-attention models.

Kundu et al. [33] proposed an unsupervised anomaly detection framework using an attention-based auto-encoder (A3D) to detect false data injection attacks. The A3D framework consists of two encoder layers followed by an attention layer to compute the optimal weight vectors of every timestep of the encoder output. The attention vectors are multiplied by the encoder output and serve as input to the decoder. The divergence of reconstruction error is utilized to detect and localize the attacks, and the threshold is set using the precision-recall curve. The used data are collected from a real-world power grid, and a simulation of the grid is created to simulate the attack scenarios. The simulated attacks include compromising 4, 8, 12, 16, 20, and random devices out of 39 measuring devices. The A3D model showed consistent performance for all the attack scenarios ranging from 0.94 to 0.99 F1-score compared to other supervised and unsupervised non-attention models.

In [34], a network intrusion detection model based on two attention mechanisms is proposed. The model consists of location-based attention, which is implemented by a fully connected layer with a softmax activation to assign weights to the input features based on their importance. Following that, two Bidirectional GRU layers are used to process the time series. Then, dot-product attention is adopted for utilizing the adjacent traffic information to predict the current. The traffic information is grouped into slices, and for each timestep, a hidden representation is computed by a single-layer perceptron. Then, the softmax function is used to obtain the attention weights, and the weighted sum is computed as the output of the attention layer. For evaluating the model, the

UNSW-NB15 dataset is used, and an improvement of 1.5% in the detection accuracy is achieved.

2.2.2. Time Series Images

The motivation of transforming time-series into images is to obtain a visual representation of the time series, which may reveal distinctive patterns that can be used to enhance the learning task.

Yang et al. [35] proposed a method of imaging MTS data into 2D images by transforming each batch of the MTS into multiple images in which each image corresponds to one variable. Then the images are appended together vertically such that each time series is treated as an independent data channel. These aggregated images are then fed into CNN for classification.

Similarly, in [36], the authors proposed an imaging MTS technique for sensors' classification by first transforming each time series into RGB colored images using GAF or MTF. Then, each image is separated into its three monochrome images so that for each mono-color, the images from all time-series are concatenated vertically into a bigger image of size $m \times s \times m$, where m is the number of time-series and s is the size of the individual images. Thus, the CNN will have 3 input channels; each corresponds to a mono-color and has a size of m x s x m.

In [37], authors used the Gramian Angular Field and Recurrence Plot [38] to forecast the availability of crowdsourced services in a specific area based on their historical availability. For prediction, the obtained images are used as input to a residual learning-based deep neural network. The local phase quantization and local binary pattern features are extracted from the images. Then, Random Forest is used for classification.

In [39], the authors analyzed the fund price time-series data to provide a fund recommendation framework by using multiple variables. A total of 16 variables were considered, which are categorized into 4 data types. For learning, intervals of 60 trading days are used, and the variables are converted to 16 GAF matrices. To fuse the variables into one image, the 16 GAF matrices were learned as heat maps which are arranged in 4x4 sequential data form using 4 different colors; each one corresponds to one data type. This arrangement results in 16 60x60 images concatenated into 240x240 images and fed as an input to the CNN.

CHAPTER 3: INTRUSION DETECTION SYSTEM BASED ON IMAGING TIME SERIES

This chapter details the design of the intrusion detection system based on time series classification for CPS using imaging time-series approach and CNN. The high-level problem formulation for this IDS is: given a multivariate time-series of CPS data, either representing the physical process readings (i.e., sensors and actuators readings) or network traffic metrics, predict whether the system is operating normally or exhibiting abnormal behavior in the corresponding time-window.

3.1. IDS-ITS Architecture

Figure 3 depicts the proposed design for the Intrusion Detection System based on Imaging Time Series (IDS-ITS) for CPS. A sliding window-based approach is used for attack detection in which the time series is divided into T time slots with an overlap l. The IDS-ITS takes as input a multivariate time series of size T representing the physical process metrics of CPS (i.e., sensors and actuators) or network traffic to detect whether the data reflect the normal behavior of the system or exhibiting cyberattacks. The IDS will make a prediction every $p = T - l$.

IDS-ITS network consists of two parts. The first part consists of two pathways. The first pathway is trained on the GAF images of the original MTS, S, of CPS attributes readings. For the second pathway, the input time series differs depending on the available data to the IDS. If the network data is available, the pathway is trained on the GAF images of the network traffic features extracted with the alignment of the process-level data. Otherwise, the pathway is trained on the GAF images of artificial MTS representing the difference time series, S_d. This MTS is constructed by the difference between every two consecutive readings of the same attribute. Each pathway consists of N 3D-CNN blocks. Each 3D-CNN block consists of 1 convolution layer with ReLU activation followed by a max-pooling layer. Then, the outputs of the pathways S_{GAF} and S_{d_GAF} are merged to obtain S_F. In the second part, the output S_F is flattened and fed into a Batch Normalization layer followed by one fully connected layer and an output layer with sigmoid activation. The network is trained to minimize the cross-entropy function that measures how far is the network prediction Y_{pred} from the actual output Y_{actual}.

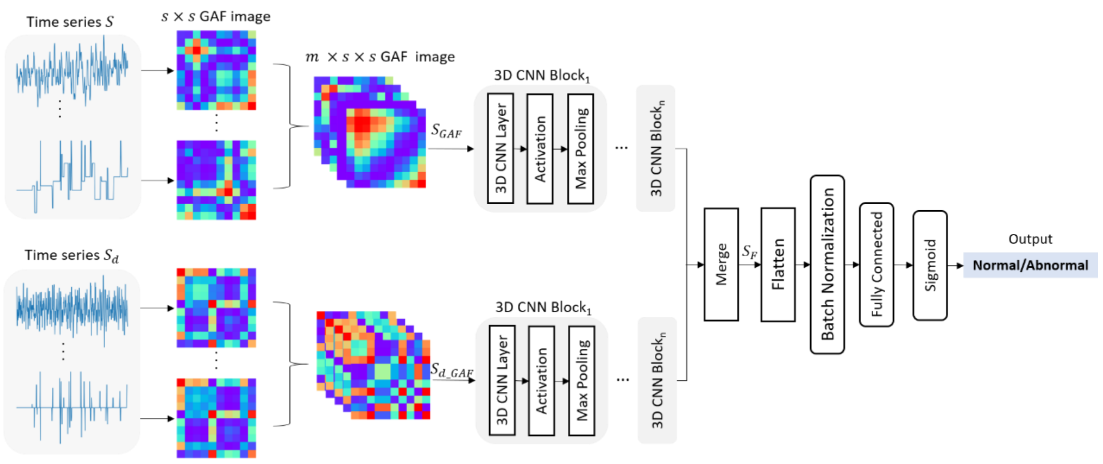

Figure 3: IDS-ITS architecture.

3.2. Time-Series Images

Our process of imaging time series is depicted in Figure 4. The Gramian Angular Field method [40] is used to obtain a visual description of the time series readings. In the following, we detail the process of generating the GAF images.

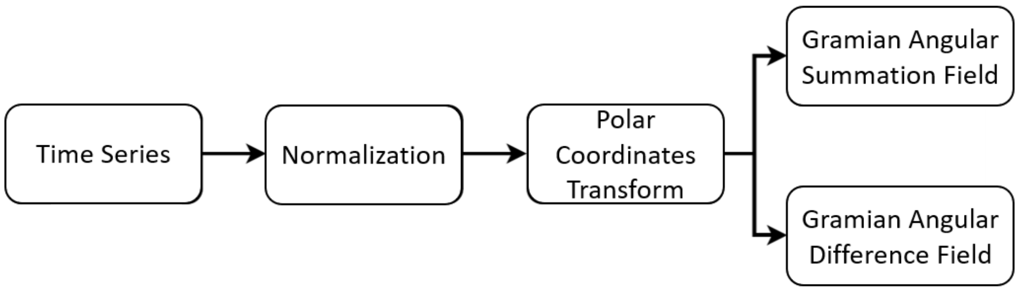
Time Series
Normalization
Polar Coordinates Transform
Gramian Angular Summation Field
Gramian Angular Difference Field

Figure 4: The process of imaging times series.

Let $\mathrm{X} = \{\mathrm{x}_1, \mathrm{x}_2, \dots, \mathrm{x}_\mathrm{n}\}$ be a univariate time series of size n and $T = t_1, t_2, \dots, t_n$ is the corresponding timestamp i.e. x_i is captured at a time t_i. The first step is to normalize X so that $\{x_i\}_{i=1\dots n}$ is in $[-1,1]$. This can be done as follows:

$$\hat{x}_i = \frac{2x_i - \max(X) - \min(X)}{\max(X) - \min(X)}$$

The scaled time series denoted by $\hat{X}$ is then transformed into polar coordinates. This can be achieved by applying the angular cosine on $\widehat{x_i}$ to obtain the angle and encoding the timestamp t_i as the radius:

$$\begin{cases} \Phi_i = \arccos(\widehat{x_i}), & \widehat{x_i} \in \hat{X} \\ r_i = \dfrac{t_i}{c}, & t_i \in T \end{cases}$$

where c is a constant to control the span of the polar coordinates. The GAF images can be constructed using Gramian Angular Summation Field (GASF) and Gramian Angular Difference Field (GADF). The motivation is to exploit the temporal correlation between the time intervals of the same attribute by trigonometric means.

GASF is a matrix of the form:

$$GASF = \begin{pmatrix} \cos(\Phi_1 + \Phi_1) & \dots & cos(\Phi_1 + \Phi_n) \\ cos(\Phi_2 + \Phi_1) & \dots & cos(\Phi_2 + \Phi_n) \\ \vdots & \ddots & \vdots \\ cos(\Phi_n + \Phi_1) & \dots & \cos(\Phi_n + \Phi_n) \end{pmatrix}$$

Each $GASF_{i,j}$ represents the temporal correlation by the summation of angular directions. The main diagonal of $GASF$ is the special case which represents the original angular information.

GADF is a matrix of the form:

$$GADF = \begin{pmatrix} \cos(\Phi_1 - \Phi_1) & \dots & cos(\Phi_1 - \Phi_n) \\ cos(\Phi_2 - \Phi_1) & \dots & cos(\Phi_2 - \Phi_n) \\ \vdots & \ddots & \vdots \\ cos(\Phi_n - \Phi_1) & \dots & \cos(\Phi_n - \Phi_n) \end{pmatrix}$$

$GADF$ reflects the temporal correlation by the difference of angular directions. Figure 5 shows the GADF and GASF images of the Level Indicator Transmitter sensor (LIT-101) from the SWaT-2015 dataset for both normal and attack instances.

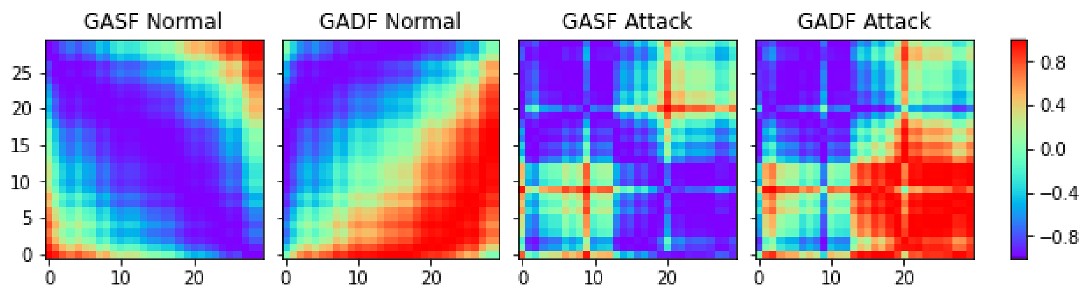

Figure 5: Gramian Angular Field of Level Indicator Transmitter sensor readings from the SWaT-2015 dataset for a window size of 30 seconds.

The Gramian matrix has a size of $n \times n$, where n is the length of the raw time series. Piecewise Aggregation Approximation (PAA) is used to reduce the size of the image, which is a dimensionality reduction method for time-series data. PAA converts the time series from length n to s by dividing the time series into s segments and replace the data points in each segment by their average value. After applying PAA, images of size $s \times s$ are obtained.

3.3. Multivariate Time-Series

The previous works for imaging MTS mostly use image concatenation to represent the images constructed from each time series into a 2D space. The main challenge of MTS classification is that the classifier has to capture the correlations and dependencies between the variables along with the features of the variables. The MTS consists of more than one time-dependent variable where each variable does not depend only on its previous values but also depends on the previous and current values of other variables. These dependencies introduce strong temporal correlations between the variables, which might not be captured by image concatenation.

Our approach of imaging the MTS aims to preserve the temporal correlations by converting the time series into 3D images to capture more fine-grained representations of the features. The usage of 3D images will help capture the time dependencies by representing all the time series in the same time-space. To construct the 3D images, each MTS transformation will result in multiple images where each one corresponds to one univariate time series. All the images have the same size of $s \times s$ where $s < n$ (n is the time series length). So, assuming there are m variables, the transformation of each time window of the MTS to GAF will produce a 3D matrix of size $m \times s \times s$ which can be perceived as a 3D image.

3.4. Network Inputs

There are two inputs to the IDS, the first input S which is the MTS of the real readings of the variables in the CPS data. The MTS is defined as $S = \{X_1, X_2, \dots, X_{m\}}$, where X_i is a univariate time series and m is the number of features in the dataset. All the univariate time series have the same length n, and the same timestamps T.

For the second input, we consider two types. The first input type represents the difference time series $S_d = \{X_{d_1}, X_{d_2}, \dots, X_{d_m}\}$, where X_{d_i} is a univariate time series representing the difference between the readings at timestamp t and the previous readings at timestamp $t - 1$ such that each point $x_{d_t} \in X_{d_i}$ is equal to $x_{d_t} = x_t - x_{t-1}$. The intuition behind the difference time series is that under normal operations of the CPS, the variations between every two consecutive readings are expected to be very small. On the other hand, in the case of cyber-attack, unexpected variations in the readings are usually exhibited.

The second type of input is the network traffic. In this case, the same notations are applied where X_{d_i} refers to the time series of a specific network feature extracted by considering the aggregated network information at the same time-slots considered for collecting the process data. For example, if the process-level data are collected every second, the corresponding network features are extracted from all the packets in one second. Then, they are aggregated by either summing or averaging to have one input sample for each corresponding sample in the process-level data. Examples of aggregated network traffic are traffic volume, average inter-arrival time, and average response time.

After constructing the input samples for both time series where each sample has a size of $m \times n$, the samples are transformed into time series images using GADF or GASF to obtain images of size $m \times s \times s$. These images are the input to the CNN network.

3.5. Merge Techniques

The IDS network consists of two similar pathways S_{GAF} and S_{d_GAF}. The outputs of the two paths are merged to obtain S_F, which has the same 3D dimensions as the individual fused inputs, but a different number of activation maps based on the used merge technique. Several merge techniques are examined, which can be categorized into simple techniques and multimodal techniques. The simple techniques include concatenation and addition. The multimodal techniques include the Compact Bilinear Pooling (CBP) and EmbraceNet.

3.5.1. Concatenation

A simple concatenation of two or more inputs where all the inputs to the concatenation layer have the same 3D dimensions. For example, assuming we have two inputs to the concatenation layer with sizes $m \times s \times s \times f_1$ and $m \times s \times s \times f_2$ where f is the number of activation maps determined by the number of filters in the previous convolution layer. Then the concatenation layer will stack the two inputs and result in one output of size $m \times s \times s \times (f_1 + f_2)$

3.5.2. Addition

Element-wise addition consists of summing all the corresponding entries from all inputs. The output of the Add layer will have the same shape as its inputs.

3.5.3. Compact Bilinear Pooling

The bilinear pooling method computes the outer product of two vectors to produce a combined representation of the vectors. Lin et al. applied the bilinear pooling for fine-grained visual recognition [41]. The architecture consisted of two feature extractors represented by CNNs. The resulting vectors were multiplied by an outer product and pooled to obtain a bilinear vector that is fully connected to an output layer. Even though a significant improvement was achieved for the visual recognition task, the outer product operation results in a remarkably high dimension representation of the bilinear vector. To tackle this inefficiency, Gao et al. introduced compact bilinear pooling that compresses the bilinear pooling efficiently for one input [42]. The compact bilinear pooling was adopted by Fukui et al. for multimodal learning for the task of visual question answering and visual grounding [43].

CBP applies the count sketch projection function Ψ to each modality by projecting the vector $v \in R^n$ to the vector $y \in R^d$. To perform the projection, two vectors are initialized randomly from a uniform distribution. The first vector $s \in \{1,-1\}^n$, which has the same length as v and contains either 1 or -1 for each index. The second vector $h \in \{1, \dots d\}^n$, maps each index in v to an index in y. To compute the values of y, first, it is initialized as a zero vector. Then for each element v_i, the corresponding element in h is found, and y is updated as:

$$y[h[i]] = y[h[j]] + (s[i] \times v[i])$$

After computing the count sketches for both modalities, the combined representation of the two modalities v_1 and v_2 is computed as:

$$\Phi(v_1, v_2) = FFT^{-1}(FFT(y_1) \odot FFT(y_2))$$

where the outer product of the two vectors is replaced by a convolution of the count sketches vectors. Then, the convolution operation is replaced by an element-wise product in the frequency domain. In our work, CBP is used to get a combined representation of the two input paths. Figure 6 shows the steps of the CBP for two inputs.

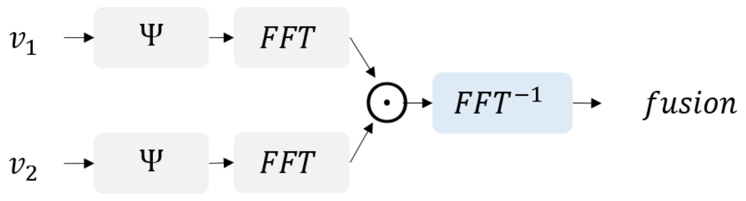

Figure 6: The steps of the multimodal compact bilinear pooling merge technique.

3.5.4. EmbraceNet

EmbraceNet [44] is a multimodal integration technique that consists of two components: the docking layers and embracement layers. The docking layers convert the input vectors to a dockable format to ensure that all vectors have the same size c. The conversion is done as:

$$d^k = f(w_i \cdot x + b_i)$$

where f, w_i, b_i, and x are the activation function, the weight vector, the bias, and the input vector. In the embracement layer, the input vectors are combined into one vector of size c. First, for each index i of the dockable vectors, a vector r_i of a length equal to the number of inputs is created from a multinomial distribution such that only one value is equal to 1 and the rest are 0. Each value in the output vector is computed as:

$$e_i = \sum_{k=1}^{n} r_i^k \circ d_i^k$$

where n is the number of inputs. The output of the embracement layer combines data from all the inputs where r determines which of the inputs contribute to each component in the output vector. Figure 7 illustrate the process of EmbraceNet with two inputs v_1 and v_2.

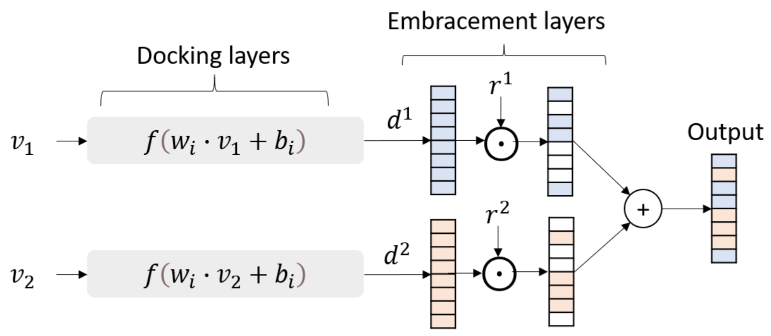

Figure 7: The steps of EmbraceNet multimodal merge technique.

CHAPTER 4: INTRUSION DETECTION SYSTEM BASED ON MULTIMODAL ATTENTION NETWORK

In this chapter, the design of the proposed Multimodal Attention Intrusion Detection System (IDS-MAN) is presented along with the details of all the constituent components. The IDS processes the inputs time series using GRU layers. Then a Multimodal Attention technique is used to capture the most important relationships between the different modalities.

4.1. IDS-MAN Architecture

The model aims to leverage the different data sources of the CPS to learn a joint representation and find some relationships between the inputs for enhancing the attack detection accuracy. Figure 8 depicts the architecture for the IDS-MAN model. The model consists of three components: the modality paths, the Multimodal Attention Network (MAN), and the output network. First, the data extracted from each source are processed independently by different modality paths. The input data sizes differ at each path, while all the outputs share the same dimensions. The modalities representations are then passed to the multimodal attention layers to learn the pair-wise joint representation between every pair of modalities. After that, the outputs from the MAN layers and the modality paths are concatenated to represent the joint global representation of all the modalities and their relationships. The joint representation is then passed to a batch normalization layer to standardize its inputs, followed by a time-distributed dense layer. The output network receives the flattened vector and passes it to a dense layer to reduce its dimension. The network is trained with joint supervision using two output layers, a softmax layer to separate the inter-class difference and a center loss layer to reduce the intra-class variations.

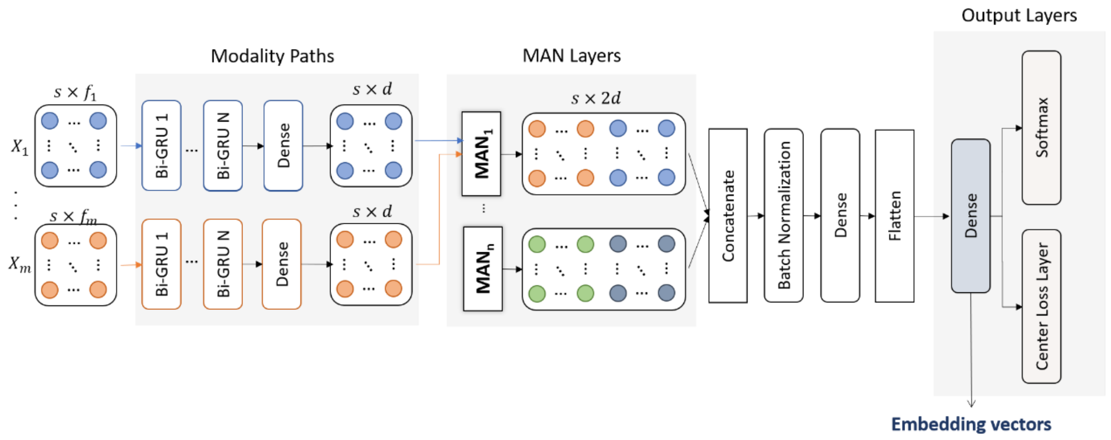

Figure 8: IDS-MAN architecture.

4.2. Modality Path

Separate network paths process the raw data from each modality before the fusion. Let $X = \{x_1, \ldots, x_m\}$ be the input to the IDS where m is the number of modalities. Then $x_i \in \mathbb{R}^{s \times f_i}$, where s is the number of timesteps that is unified for all the modalities, and f_i is the number of features in the corresponding modality. Each modality path has N GRU layers followed by one Dense layer. The output of the GRU layer can be represented by $R_i \in \mathbb{R}^{s \times u}$, where u is the number of units in the GRU layer. The output of the last GRU is passed to a time-distributed dense layer to process each temporal slice. The dense layer result in $D \in \mathbb{R}^{s \times d}$, where d is the number of units in the dense layer. For our model, we used bidirectional GRU (Bi-GRU) that consists of two GRUs. The first one processes the input in a forward direction where the state of each timestep is determined by the past timesteps. The second one processes the data in a backward direction in which the state of each timestep is determined by the future timesteps. The output is the concatenation of the forward and backward GRUs.

4.3. Multimodal Attention Learning

For the design of our IDS, we adopted the Multimodal Attention technique followed by [13]. The MAN layer receives the modalities' representations and learns the joint representation for every pair of modalities. Figure 9 shows the inner computations for the MAN layer. Assuming we have two sources of data, D and T. Then the modalities representations are $D \in \mathbb{R}^{s \times d}$ and $T \in \mathbb{R}^{s \times d}$ where s is the timesteps and d is the dimension of dense layer output. First, the cross-modality representations M_1 and M_2 are computed by the dot product. Then, the attention weights for the M_1 and M_2 are calculated by the softmax function to assign probability scores to every timestep.

$$M_1 = T^T \cdot D \text{ and } M_2 = D^T \cdot T$$

$$\begin{aligned} N_1 &= \frac{e^{M_1(i,j)}}{\sum_{k=1}^{s} e^{M_1(i,k)}} \\ & \qquad\qquad\qquad for\ i,j = 1, \dots, s \\ N_2 &= \frac{e^{M_2(i,j)}}{\sum_{k=1}^{s} e^{M_2(i,k)}} \end{aligned}$$

The intuition behind the dot product operation is to find the association between the feature vectors of every two timesteps in the two modalities. The attention weights are computed to emphasize the more contributing associations. After that, the modality-wise attentive representations are calculated as follows:

$$O_1 = N_1 \cdot T \text{ and } O_2 = N_2 \cdot D$$

Multiplying the attention weights by the modality representation implies that the modalities will have different contributions to the final representation. Finally, element-wise matrix multiplication is computed to model the interaction between the multimodal specific representation of each modality (i.e., O) and the other modality representation. Then, the concatenation of the two matrices accounts for the output of the MAN layer.

$$A_1 = O_1 \odot D \text{ and } A_2 = O_2 \odot T$$

$$Output(\text{MAN}) = concat[A_1, A_2]$$

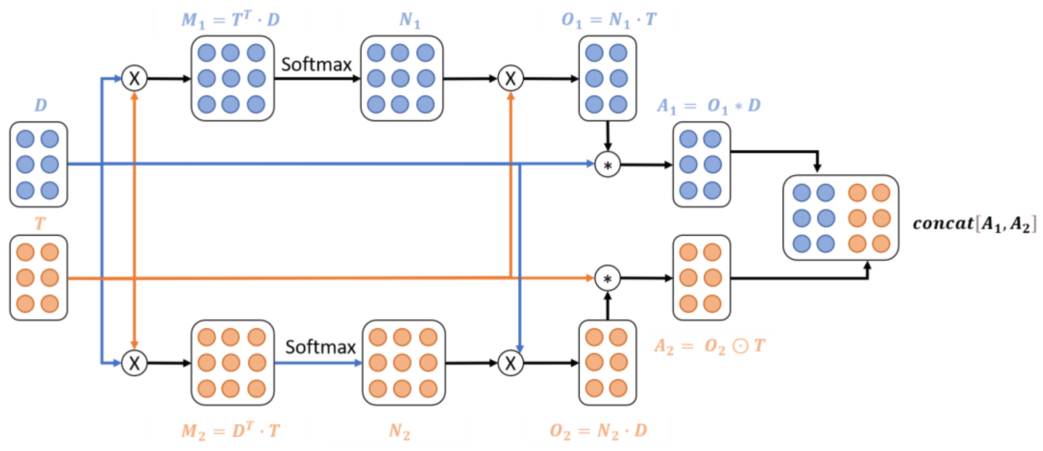

Figure 9: The multimodal attention computations.

After obtaining the pair-wise joint representations for every pair of modalities, all the outputs of the MAN layers are concatenated together with the modality paths' outputs obtaining a joint global representation. The output of the concatenation is $F \in \mathbb{R}^{s \times d^c}$ where d^c is computed as:

$$d^c = s \times d\big(m + 2C(m, 2)\big).$$

$$C(m, 2) = \frac{m!}{(2!\,(m-2)!)}$$

After that, a Batch Normalization layer is used to stabilize the network's training, followed by a time-distributed dense layer to perform the same processing to each timestep. The output of the dense layer is then passed to a flatten layer.

4.4. Output Network

The output network consists of three layers. The first one is a dense layer to reduce the size of the flattened vector. The output of this layer serves as the vector representation of the input which is used in the detection. It can be represented as $V \in \mathbb{R}^{d^v}$, where d^v is the number of units in the dense layer. The second layer is a softmax layer to normalize the output to a probability distribution. The third layer is a center loss layer which is used to learn embedding representations of the classes. The network is trained using joint supervision between the softmax loss L_s and the center loss L_c. The total loss is given as:

$$L = L_s + \lambda L_c$$

where λ is a scalar used to balance between the two losses.

4.4.1. Center Loss

The center loss is proposed in [45] to enhance the discriminative power of the learned features in deep neural networks and decrease the variance of the intra-class features. The intuition behind the center loss is to learn a center of deep features for each class and train the network to minimize the distance between the deep features and their class center. In this work, we adopt the same computations of the center loss but extend the deep feature's dimension to learn embedding vectors for the classes instead of centers. The center loss function is defined as follow:

$$L_c = \frac{1}{2}\sum_{i=1}^{b}\left\|x_i - v_{y_i}\right\|^2$$

where b is the batch size and v_{y_i} denotes the embedding vector of the class y_i. Typically, the class embedding vectors should be updated every epoch; however, because the network is trained using mini-batches of size b, a scalar parameter α is introduced to control the learning rate of the embedding at each iteration. The update equation of the embedding vector is the same as the update equation for the centers proposed in the original paper, which is represented as follow:

$$v_j^{t+1} = v_j^t - \alpha \Delta v_j^t$$

$$\Delta v_j = \frac{\sum_{i=1}^{b} v_j - x_i \in j}{1 + \sum_{i=1}^{b}(y_i = j)}$$

4.5. Attacks Detection

After training the network to produce embedding vectors for the classes, the cosine similarity is used to measure the similarity between the learned classes' embedding vectors v_j and the vector of the input's deep learned features x_i. The cosine similarity between v_j and x_i is calculated as:

$$\cos(\theta) = \frac{v_j \cdot x_i}{\|v_j\| \|x_i\|}$$

Then the classification is determined based on the class with the highest similarity.

CHAPTER 5: INTRUSION DETECTION SYSTEM EVALUATION

5.1. CPS Datasets

There is a lack of available CPS datasets that provides multimodal data from different CPS layers. Thus, we used the SWaT-2015 dataset, which contains process-level data from different physical process stages. In addition, we used the SWaT-2019 dataset, which is a small-scale dataset containing process-level and network-level data

5.1.1. The Secure Water Treatment Datasets

For evaluating the proposed models, two datasets constructed from the Secure Water Treatment (SWaT) fully operational testbed are used [46]. The process of SWaT has six stages. In the first stage, the raw water is stored in a tank, then passed to the second stage to assess the water quality. In the third stage, the water is filtered to remove unwanted materials, followed by a dechlorination process to remove any remaining chlorine in the fourth stage. After that, the water goes through the reverse osmosis system to reduce the inorganic impurities. At the last stage, the water is stored and prepared for distribution. Each stage in the process contains multiple sensors and actuators. Besides the physical process, the SWaT testbed communication is designed to resemble the real communications in CPS, consisting of a layered communication network of two levels, programmable logic controllers, human-machine interfaces, and a historian data server. The used datasets are collected using different data collection setups in 2015 and 2019.

5.1.1.1. SWaT 2015

This dataset contains data from 24 sensors and 27 actuators across the six stages of the process at the physical process level. The data were collected for 11 days, considering the testbed's normal operation for 7 days and several attack scenarios for the remaining 4 days. A total of 36 attack scenarios were conducted on different sensors and actuators categorized into single-stage single-point, single-stage multi-point, multi-stage single-point, and multi-stage multi-point. Each attack lasted between few minutes to several hours. During this period, the sensors' and actuators' data were continuously recorded every second in the historian server, resulting in a dataset containing 946,722 samples for 51 attributes. Table 1 presents the attack scenarios in the SWaT-2015 dataset.

Table 1: Attack Scenarios in SWaT-2015 Dataset.

Attack#	**Attack Point**	**Attack Scenario**
1	MV-101	Open MV-101 to cause tank overflow
2	P-102	Turn on P-102 to cause pipe bursts
3	LIT-101	Increase by 1 mm every second to damage P-101 and cause tank underflow
6	AIT-202	Set AIT-202=6 to cause P-203 to turn off and change the water quality
7	LIT-301	The water level increased above HH, which causes the stop of the inflow.
8	DPIT-301	Set DPIT >40kpa, which causes a decrease in the water level of tank 401 and an increase in the water level of tank 301
10	FIT-401	Set FIT-401 <0.7 to shutdown the UV.
11	FIT-401	Set FIT-401=0 to shutdown the UV.
13	MV-304	Close MV-304 to halt stage 3.
14	MV-303	Do not let MV-303 open to halt stage 3.
16	LIT-301	Decrease water level by 1mm each second to cause tank overflow.
17	MV-303	Do not let MV-303 open to halt stage 3.
19	AIT-504	Set AIT-504=16 uS/cm, which causes the RO to shutdown and water go to drain.
20	AIT-504	Set AIT-504=255 uS/cm, which causes the RO to shutdown and water go to drain.
21	MV-101, LIT-101	Keep MV-101 on and set LIT-101=700 mm to cause tank overflow
22	UV-401, AIT-502, P-501	Stop UV-401, set AIT502=150, and force P-501 to remain on to damage the RO.
23	P-602, DIT-301, MV-302	Set DPIT-301>0.4 bar, keep MV-302 open, and keep P-602 closed to cause a system freeze.
24	P-203, P-205	Turn of P-203 and P-205 to change water quality

Attack#	Attack Point	Attack Scenario
25	LIT-401, P-401	Set LIT-401=1000 and keep P402 on to cause tank underflow.
26	P-101, LIT-301	P-101 is turned on continuously and set LIT-301=801 mm, which causes tank 101 underflow and tank 301 overflow
27	P-302, LIT-401	Keep P-302 on, and set LIT401=600 mm for 9 minutes to cause tank overflow
28	P-302	Close P-302 to stop the inflow of tank T-401
29	P-201, P-203, P-205	Turn on P-201, P-203, and P-205 to cause wastage of chemicals
30	LIT-101, P-101, MV-201	Turn on P-101 and MV-101, set LIT-101=700 mm, P-102 started itself because LIT301 level became low. This cause tank 101 underflow and Tank 301 overflow
31	LIT-401	Set LIT-401 to less than L to cause tank overflow
32	LIT-301	Set LIT-301 to above HH to cause tank underflow and damage P-302
33	LIT-101	Set LIT-101 to above H to cause tank underflow and damage P-101
34	P-101	Turn P-101 off to stop the outflow.
35	P-101; P-102	Turn P-101 off and keep P-102 off to stop the outflow
36	LIT-101	Set LIT-101 to less than LL to cause tank overflow
37	P-501, FIT-502	Close P-501 and set FIT-502=1.20 after 90 seconds to reduce the output.
38	AIT-402, AIT-502	Set AIT402=260 and AIT502=260 to cause the water to go to the drain because of overdosing
39	FIT-401, AIT-502	Set FIT-401=0.5 and AIT-502=140 mV to shutdown UV, and water will go to RO.
40	FIT-401	Set FIT-401=0 to shutdown UV, and water will go to RO.
41	LIT-301	Decrease value by 0.5 mm per second to cause tank overflow

5.1.1.2. SWaT 2019

This dataset considered two modalities of data: the physical process and network traffic. The duration is 3 hours and 40 minutes, which results in 13211 samples for the process data covering 81 sensors and actuator values. The raw network traffic is provided in pcap files capturing all the network activities of the testbed. The dataset contains two attack scenarios. For the first attack, the historian data are exfiltrated by injecting malware into the SCADA workstation using a USB thumb drive. The attack has 4 instances with a duration of 5 minutes for each one. In the second attack, the SCADA workstation is infiltrated by the command and control (C2) malware to disrupt the sensor and actuator readings. The attack has 5 instances with a duration of 3 minutes each. The dataset does not provide the details of the attacks' targets and their impact.

For our experiments, we extracted several protocol-based features from the network traffic. The timestamps of the network features are aligned with the recorded process features. Also, a time window of 1 second is used since the process features are recorded every 1 second. For each protocol, 6 features are extracted at the packet level. Then the feature's values are either averaged or summed for the duration of the time window. Along with the protocol-based features, 3 other features are extracted to describe the overall network traffic characteristics for the time window duration. The total number of features is 207 with 34 different protocols. Table 2 present the extracted features and their description. For this dataset, we have two modalities: the process data $P \in \mathbb{R}^{s\times 81}$, and the network data $N \in \mathbb{R}^{s\times 207}$ where s is the size of the time window.

Table 2: The Network Features for SWaT 2019 Dataset.

Features		**Description**
Overall	Average time-delta	The average time between every two consecutive packets.
	Total data size	The total size of the exchanged data.
	Total volume	The total number of packets.
Protocol-based	TCP stream average delta-time	The average time between every two consecutive packets in the same TCP stream.
	Average response time	The average time taken by the request to reach the destination and acknowledged by the sender
	Average frame length	The average frame length of all the packets for the protocol
	Total data size	The total size of the exchanged data across the network for the protocol
	Total frame length	The total frame length of all the packets for the protocol
	Total volume	The total number of packets with the same protocol

5.2. Evaluation Metrics

To evaluate the performance of the proposed models, we used precision, recall; also called sensitivity or true positive rate (TPR), specificity; also known as the true negative rate (TNR), F1-score, and accuracy.

$$Precision = \frac{TP}{TP + FP}$$

$$Recall\ (TPR) = \frac{TP}{TP + FN}$$

$$Specifity\ (TNR) = \frac{TN}{TN + FP}$$

$$F_1 = \frac{2 \times precision \times recall}{precision + recall}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

where true positive (TP) refers to the correctly detected attacks, true negative (TN) refers to the normal samples that are classified correctly, false positive (FP) indicates having a normal sample that is misclassified as an attack, and false negative (FN) is when an attack sample is misclassified as normal. The accuracy and F1-score show the overall performance of the IDS. Thus, the hyperparameters tuning will be based on those two metrics. In addition, the recall for each attack is recorded for the sake of comparing our results with previous works.

5.3. Data Preprocessing

The SWaT-2015 dataset contains 946,722 samples with a distribution of 94% for the normal class and 6% for the attack class. Selecting the entire samples of the dataset would introduce a huge imbalance between the normal and the attack samples, and the prediction would be biased towards the normal class. Hence, in our experiment, only the 449,921 samples that are recorded under the attacks are considered. For the IDS-ITS model, the data is further balanced by performing undersampling for the normal samples and oversampling for the attack samples. On the other hand, for the IDS-MAN model, imbalanced data is used.

The datasets contain binary and continuous features with different ranges for each feature. Thus, feature normalization is needed to ensure that all features have the same scale. Feature normalization makes the learning process easier by ensuring that all the features have the same significance in training and avoid having the results skewed toward a particular feature. Standard normalization is used for the SWaT-2015 dataset in which the data are standardized by removing the mean then scaling to the unit variance. For the SWaT-2019 dataset, the min-max normalization is used, which scales the data between [0,1].

For data splitting, we used an 80%-20% train-test split with cross-validation. In the SWaT-2015 datasets, the attacks are not repeated, so the testing is performed on unknown attacks for each fold. For the SWaT-2019 dataset, the splitting ratios vary for each split to ensure no attack instance span over the two splits.

5.4. Experimental Settings

The proposed models are implemented using the TensorFlow framework high-level API, Keras. Our networks are created using Keras functional API since they have non-linear topologies and more than one input. The evaluation for the SWaT-2019 was carried using the Google Collaboratory platform with Google compute engine backend. For the SWaT-2015, the evaluation was performed on a Quad GPU Desktop with RTX 6000. For evaluation, 5-fold cross-validation is used to test the models' performance on all the attack scenarios.

5.5. IDS-ITS Evaluation Results

5.5.1. Model Parameters

For the IDS-ITS model, the hyperparameters that need tuning are the window size and the sliding period. Besides, the network's hyperparameters which are the number of CNN blocks, the CNN layers' parameters, and the regularization values.

5.5.1.1. Window Size

To convert the time series to images, a window-based approach is used for attack detection. The choice of the time window size plays a major role in the performance of the anomaly detection system. The window size determines the most suitable period that reflects the system's behavior to detect the corresponding label for this period. Figure 10 shows the performance of the different window sizes ranging from 10s to 60s using a sliding window of $window\_size/2$. For the SWaT-2015 dataset, smaller time windows tend to have better performance in which the time windows of 10s and 30s achieved the same F1-score. However, the lowest recall is obtained by the 10s window. Thus, for the SWaT-2015 dataset, we choose the time window of 30s, which provided the best balance between the precision and the recall. For SWaT-2019, we choose the time window of 40s, which obtained the highest F1-score.

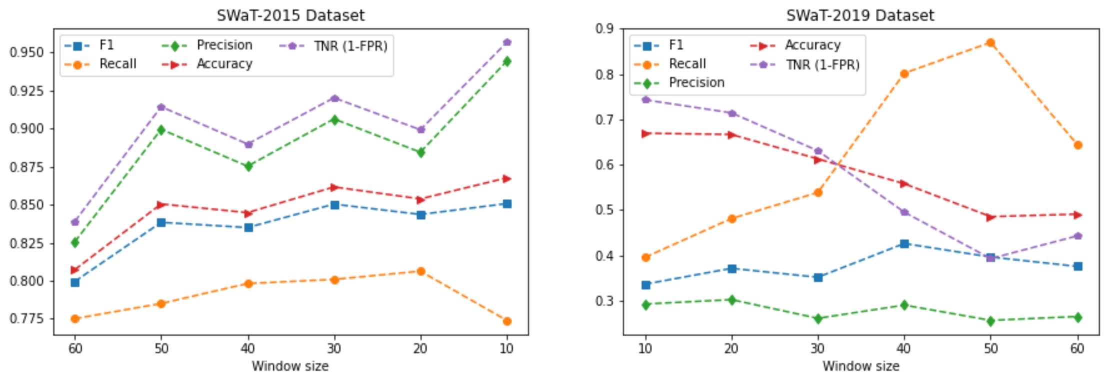

Figure 10: The evaluation metrics vs. window size ranging from 10s to 60s for the IDS-ITS model.

After choosing the most suitable window size, we tune the period of the window slide. Figure 11 shows the model's performance with a sliding window of values ranging from 1s to 30s for the SWaT-2015 dataset. A sliding window of 1s means that the model will make a prediction as soon as a new sample arrives because in the used datasets, the data are collected every second. On the contrary, a sliding window of 30s means there will be no overlap between the samples, and only one prediction will be performed every 30s. For the SWaT-2015 dataset, the model trained with a sliding window of 1s obtained the best performance in terms of precision, recall, F1-score, accuracy, and TNR. For SWaT-2019, we used a sliding window of 1s to increase the number of samples since the dataset is small.

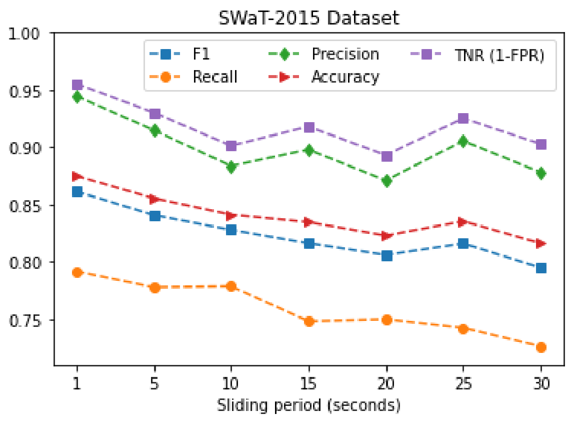

Figure 11: The evaluation metrics vs. sliding period ranging from 1s to the window size for IDS-ITS

5.5.1.2. Network Parameters

The network's hyperparameters are tuned to achieve a balance between overfitting and underfitting. The hyperparameters of the IDS-ITS network include the number of CNN blocks, the CNN layers' parameters such as the number of filters and the kernel size, the activation function, the regularization value, and the learning rate.

Table 3 provides the tuned values. For training, we used Adam optimizer with an initial learning rate of 0.001. Moreover, to avoid overfitting, we used an early stopping condition to stop the training when the loss does not improve 3 epochs.

Table 3: The Results of Hyperparameter Tuning for IDT-ITS.

Hyperparameter	**Value**
CNN Layer 1	32
CNN Layer 2	64
CNN Layer 3	128
CNN Kernel	3
Dense Layer	256
Activation	ReLU
L2 Regularization (hidden layers)	0.01
L2 regularization (output layer)	0.3 (SWaT-2015), 0.01 (SWaT-2019)
Initial Learning Rate	0.001

5.5.2. Merge Techniques

To compare the different merge approaches, similar models are trained with the different techniques. For SWaT-2015, the two inputs have the same dimension, so the 4 merge techniques are used. On the other hand, for SWaT-2019, the images of the two modalities have different dimensions because each has a different number of features. So, to map the inputs to the same dimension before merging the two modalities, the images are flattened and passed to a dense layer. Thus, only three merge techniques are used since the CBP requires the input to be in image format. Figure 12 shows the performance of the different models trained using the 4 merge methods along with models that are trained using only one input. We refer to compact bilinear pooling as CBP, concatenation as CC, addition as Add, EmbraceNet as ENet. For SWaT-2015, OP refers to a one-path model trained on the original time-series images. For SWaT-2019, PP refers to the model trained using process-level data, and NP refers to the model trained with the network-level data.

For SWaT-2015, ENet achieved the best performance in terms of F1-score, accuracy, precision, and TNR, followed by CBP by a difference of 0.5% in the accuracy and F1-score. On the other hand, the CC obtained the highest recall but a lower F1-score and accuracy by 8% compared to the ENet. Moreover, the models with two paths show an improvement of 0.3%, 0.6%, 8%, and 9% for Add, CC, CBP, and ENet, respectively, in F1-score compared to the OP model.

For SWaT-2019, ENet obtained the best performance for all the evaluation measures, followed by CC with accuracies of 79% and 76% and F1-scores of 56% and 51%. On the other hand, the PP and NP models showed significantly lower F1-scores falling behind the ENet model, with 29% and 23% differences. The Add model achieved a slightly better F1-score compared to the PP and NP models. However, it obtained the lowest accuracy among all the models.

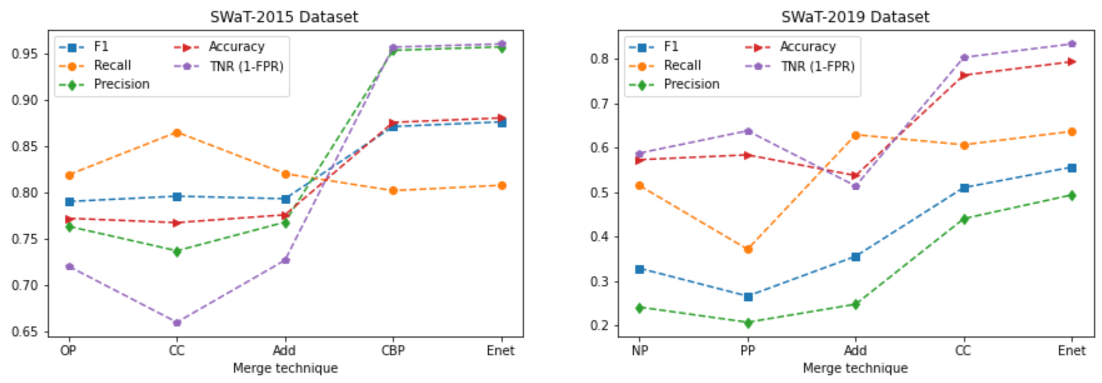

Figure 12: The evaluation metrics for the different merge techniques for the IDS-ITS model.

5.5.3. Input Paths for SWaT-2015

The last column in Table 6 shows the recall values for each attack for five different variations of IDS-ITS for the SWaT-2015 dataset. To emphasize the significance of the difference time series S_d, the first model is trained using only one path with the original time series S while the other four are trained using the two paths with different merge techniques. The OP model achieved a better detection rate for 11 attacks compared to the average results of the two-path models, with an increase ranging from 1% to 37%. It is worth noting that most of the attack that obtained remarkably better results with the OP model target actuators with discrete values. This is because, for any variations in the actuator's readings, the values of the difference between every two consecutive readings are discrete. In the case of attacks, the variations in the difference values only occur at the instances that indicate the start and the end of attacks. All other instances in which the values do not change have values equal to 0. Thus, adding the second network path that is trained on the difference time series does not contribute significantly in distinguishing these attacks since most of the records are zeros. For the two-path models, most of the attacks that obtained a remarkably higher detection rate target sensors with continuous values with small consistent variations in their readings. So, in general, for normal instances, the difference time series values are not constant.

5.6. IDS-MAN Evaluation Results

5.6.1. Model Parameters

For the IDS-MAN model, three hyperparameters need to be tuned: the window size, the dimension of embedding vectors for the classes, and the learning rate α for the embedding update. Moreover, all the network's hyperparameters need to be tuned, including the number of GRU layers, the number of units, and the regularization values.

5.6.1.1. Window Size

The window size refers to the number of timesteps to be considered for each input to the network. To tune the window size, we choose to compare between the sizes of 120, 90, 60, and 30 seconds with a sliding window of 5 seconds for SWaT-2015 and 1 second for SWaT-2019. Figure 13 shows the results for both datasets. For SWaT-

2015, we choose a window size of 30s, which achieved a slightly higher F1-score and TNR. For SWaT-2019, the window of size 60s obtained the best F1-score, accuracy, precision, and TNR.

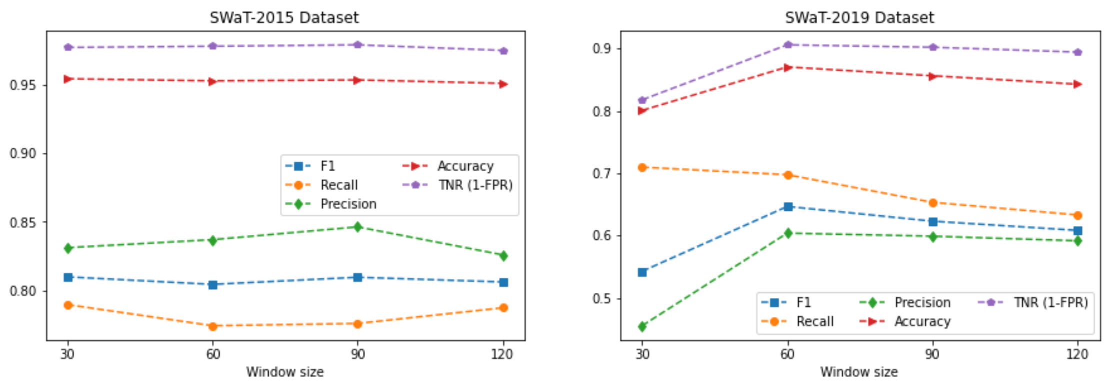

Figure 13: The evaluation metrics vs. window size ranging from 30s to 120s for the IDS-MAN model.

After choosing the window size, we tune the period of the window slide. Figure 14 shows the model's performance with a sliding window of values ranging from 1s to 30s for the SWaT-2015 dataset. The model trained with a sliding window of 5s obtained the best performance in the recall, F1-score, and accuracy. For SWaT-2019, we used a sliding window of 1s to increase the number of samples since the dataset is small.

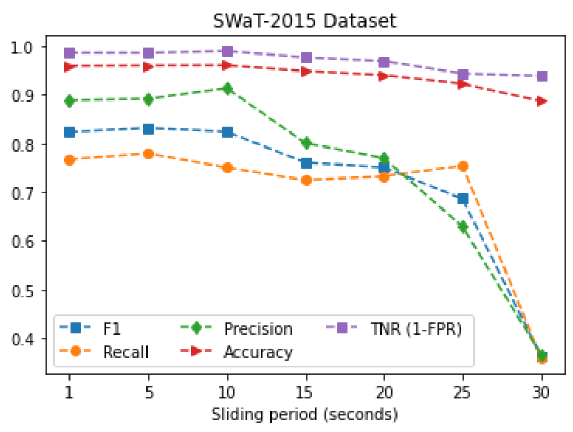

Figure 14: The evaluation metrics vs. sliding period ranging from 1s to the window size for IDS-MAN.

5.6.1.2. Embedding vectors dimension

Figure 15 shows the evaluation results for different dimensions of the embedding vectors. The dimension is tuned for the values 32, 64, 128, 256, and 512 for SWat-2015 and 2, 8, 16, 32, and 64 for SWaT-2019. For SWaT-2015, the dimension of 32 achieved the best precision, TNR, and F1-Score, but the lowest recall. On the other hand, the best recall is obtained by the 256 dimension. Accordingly, we choose the dimension of 128, which provided the best balance between the precision and recall obtained by the other dimensions. For SWaT-2019, the embedding dimension of 32 achieved the best F1-score, accuracy, precision, and recall.

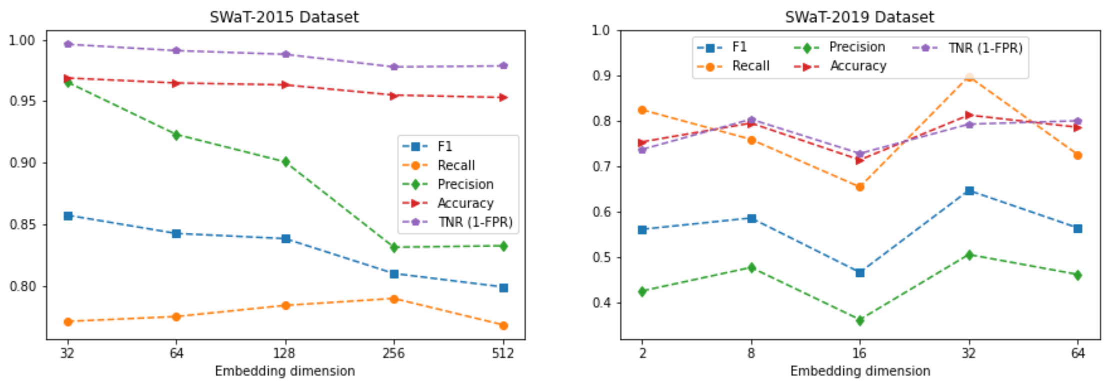

Figure 15: The evaluation metrics vs. embedding dimensions for the IDS-MAN model.

5.6.1.3. Learning Rate for the Embeddings

The learning rate α controls the update of the embedding vectors at each iteration to avoid the large change that could happen due to unrepresentative batches. Figure 16 shows the IDS's performance for α ranging from 0.1 to 1 with a step of 0.1. As shown from the figure, for SWaT-2015, the values of F1-score, recall, TNR, and accuracy are stable for α between 0.1 to 0.6. On the other hand, as α increases, the performance exhibits an increase in the recall value with the cost of the degradation of all other metrics. For SWaT-2019, no noticeable trend is observed. We choose $\alpha = 0.4$, which provided a balance between the TNR and TPR for both datasets.

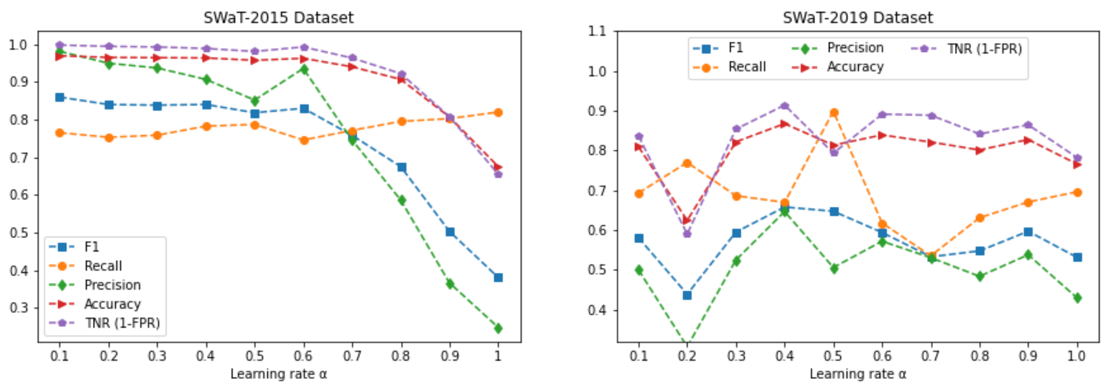

Figure 16: The evaluation metrics vs. learning rate α for the IDS-MAN model.

5.6.1.4. Network Parameters

The hyperparameters of the IDS-MAN network are the number of GRU layers, the number of units in the GRU and dense layers, the activation functions, the regularization value, and the learning rate. Table 4 provides the tuned values for the two datasets. For training, we used Adam optimizer with an initial learning rate of 0.001. Moreover, to avoid overfitting, we used early stopping to stop the training when the loss does not show improvement for 10 epochs and return the weights of the best epoch.

Table 4: The Results of Hyperparameter Tuning for IDS-MAN.

Hyperparameter	**Value (SWaT-2015)**	**Value (SWaT-2019)**
GRU Units 1	32	300
GRU Units 2	64	-
GRU Dropout	0	0.5
Dense Units (before MAN)	64	128
Dense Units (after MAN)	128	128
Dense Dropout	0	0.7
Activation	LeakyReLU	Tanh
L2 Regularization	0.8	0.1
Initial Learning Rate	0.001	0.001

5.6.2. Merge Techniques

For the SWaT-2015 dataset, we consider each stage of the process as a separate data source. On the other hand, for SWaT-2019, two different data sources are available, which are the physical process and the network traffic.

To evaluate the significance of using different data sources for the IDS, we build three different models for SWaT-2019. Two models use data from one source, while the third model uses the two modalities. The model with two modalities uses the multimodal attention network (MAN). For the models with one modality, MAN is replaced by Self Attention (SA). Figure 17 shows the evaluation measures for the three models. The model with two modalities achieved a remarkably higher F1-score with an increase of 22% and 38% for the process and network models. Moreover, significant improvements are shown in the accuracy, precision, and TNR. Meanwhile, the recall remains relatively constant for the three models.

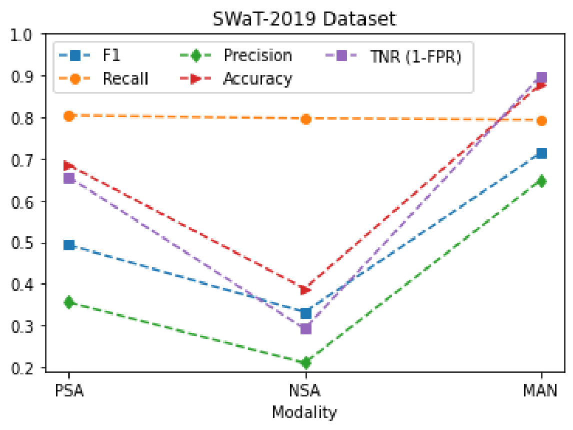

Figure 17: The evaluation metrics for the IDS-MAN model and the two SA models.

To evaluate the effectiveness of using multimodal attention to combine the modalities, we constructed two models using simple merging techniques, which are concatenation and addition. Figure 18 shows the performance of the two merge techniques and the IDS-MAN. The IDS-MAN obtained the best performance for all the evaluation measures with an F1-scores of 71%, followed by CC 62% and Add 50% for SWaT-2019. The same sequence is observed with SWaT-2015 with F1-scores of 87%, 75%, and 62%.

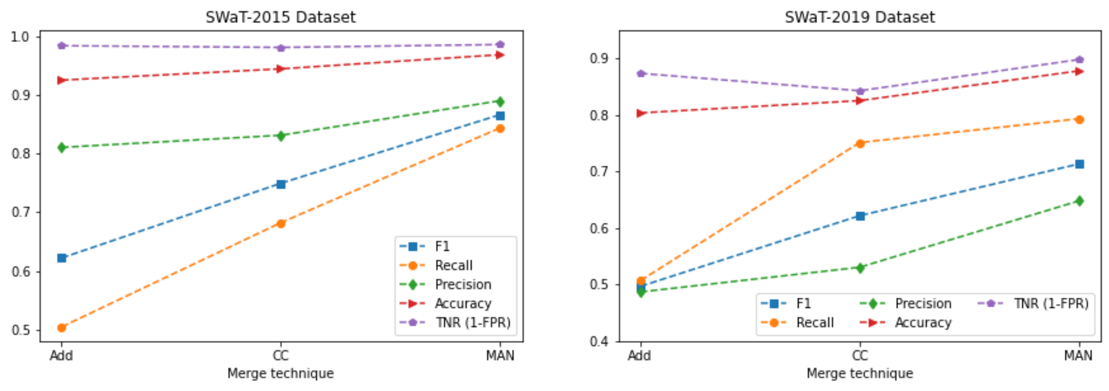

Figure 18: The evaluation metrics for the IDS-MAN and different merge techniques.

5.7. Comparison with Related Work

5.7.1. Evaluation using SWaT-2015 Dataset

IDS-ITS is able to detect all the attacks with an average detection rate of 82%. The IDS-MAN achieved a better detection rate of 84%; however, two attacks were not detected. Some attacks were harder to detect due to the nature of the attack target. IDS-ITS obtained low detection rates for attacks 13 and 14, while attack 14 is not detected by IDS-MAN. Those attacks target the same type of actuators in the same process: the motorized valves MV-303 and MV-304. The two attacks aimed to halt the operation of process 3. However, as mentioned in the dataset description, unexpected outcomes occurred, and the intent of the attacker was not achieved. Thus, these attacks were harder to detect since no effects are shown on other variables in the same process. Moreover, attacks 6, 19, 20, and 38 got low detection rates with IDS-ITS. Meanwhile, for IDS-MAN, attacks 3, 19, 31, 33, and 38 obtained low detection rates. All of those attacks did not cause an actual change in the process.

The data types of the attack targets affect the detection of the attacks. Generally, all the attacks that target the motorized valves (MV) and pumps (P), which are actuators with discrete values, have a lower detection rate than the average. In IDS-ITS, the average detection rate for the attack with discrete value targets is 27% compared to 68%

for the attacks that have at least one target with continuous values. This behavior is only observed with the IDS-ITS since small transitions in the actuators' values are exhibited under attacks. So it was harder for the network to detect the patterns of the instances for these attacks. An exception to that is attack 28, which lasted for more than 9 hours. This suggests that the attack period also affects the detection rate of the attacks. The results show that, generally, the attacks that span more than 30 minutes have a higher detection rate with an average of 69% and 62% compared to an average detection rate of 48% and 40% for attacks with a period of less than 30 minutes for the IDS-ITS and IDS-MAN respectively. Since the attack period determines the number of instances used for training the model, attacks with larger periods will have higher chances of getting recognized.

Another reason that affects the detection rate is the variance of the attack targets. The attacks with level sensors (LIT in SWaT) as the attack targets have significantly higher detection rates. It was observed that the values of these sensors have a very high variance. This observation also shows that variables with discrete values have lower detection rates since those variables typically tend to have low variance. This behavior is exhibited only with the IDS-ITS because of the characteristic of the imaging time-series approach. GAF images capture the temporal correlations between every two points by the difference or the summation of their angular directions. So, having low variance results in images with similar patterns for normal and attack instances since the values do not change frequently, and the angular cosine will be equal for many of the values. On the other hand, for variables with high variance, different patterns are obtained from the normal and attack instances since the attacks set the variables to values beyond the normal range. Hence, the angular cosine values belonging to the normal and attack instances are different, producing GAF images that can be more distinguishable.

The IDS-MAN showed superior performance for detecting the multi-points attacks that target more than one sensor or actuator either at the same stage or at different stages. The average detection rate for multi-point attacks is 72% compared to 52% for single-point attacks. The contrary is observed for the IDS-ITS in which the single-point attacks obtained an average detection rate of 58% compared to 49% for the multi-point attacks. This result suggests that the IDS-MAN is able to provide better discoverability of some of the attacks by capturing the correlations between the sensors and actuators at the different stages of the process.

The performance evaluation results for the SWaT-2015 dataset are summarized in Table 5. The accuracy and TNR are recorded; however, most related works did not include these measures. Moreover, Table 6 shows a comprehensive comparison with previous works in terms of the recall values per attack. The works for DNN, SVM, TABOR, 1D CNN, MLP, MAD-GAN, and 1D CNN-IF are described in section 2.2.1.1. The DNN and SVM [24] are considered as the baseline since they are the first work that used this dataset for IDS. The comparison shows that the IDS-ITS model achieved better results for 12 attacks. Moreover, it took the lead in detecting attacks 13 and 14, which are not detected by any previous work. On the other hand, the IDS-MAN achieved better results for 7 attacks compared to the related work and for 4 attacks compared to the IDS-ITS. Moreover, the IDS-MAN obtained the highest accuracy and TNR.

Table 5: Comparison Between Different Detection Methods for SWaT-2015 Dataset.

Method		**Precision**	**Recall**	**F1**	**Accuracy**	**TNR**	**Detected Attacks**
DNN [24]		0.983	0.678	0.803	-	-	13
SVM [24]		0.925	0.699	0.796	-	-	20
TABOR [29]		0.862	0.788	0.823	-	-	22
1D CNN 1 [22]		0.968	0.791	0.871	-	-	-
1D CNN 2 [22]		0.867	0.854	0.860	-	-	30
MLP [23]		0.967	0.696	0.812	-	-	25
MAD-GAN [26]		**0.990**	0.637	0.770	-	-	-
1D CNN-IF [28]		**-**	-	-	0.920	-	26
IDS-ITS	**CC**	0.737	**0.865**	0.796	0.767	0.660	**35**
	Add	0.768	0.820	0.793	0.776	0.727	34
	CBP	0.953	0.802	0.871	0.875	0.956	**35**
	Enet	0.957	0.807	**0.876**	0.880	0.960	**35**
IDS-MAN		0.890	0.844	0.866	**0.968**	**0.986**	33

Table 6: Recall Comparison for the Individual Attacks Between Previous Works and the IDS-ITS Model with the Different Merge Techniques and the IDS-MAN Model for SWaT-2015 Dataset.

						IDS-ITS				IDS-
#	SVM	TABOR	CNN	MLP	OP	**CBP**	**CC**	**Add**	**Enet**	MAN
1	0.00	0.05	**1.00**	0.00	0.01	0.03	0.19	0.17	0.03	0.14
2	0.00	0.93	**1.00**	0.76	0.02	0.12	0.30	0.22	0.02	0.94
3	0.00	0.00	0.23	0.00	0.18	**0.79**	**0.86**	**0.82**	**0.74**	0.07
6	0.72	**1.00**	0.90	0.95	0.08	0.11	0.50	0.32	0.08	0.96
7	0.89	0.00	**1.00**	0.91	0.92	0.96	0.97	0.97	0.96	0.96
8	0.92	0.61	**1.00**	0.98	0.10	0.16	0.72	0.59	0.18	0.97
10	0.43	0.99	**1.00**	0.98	0.58	0.94	**1.00**	0.98	0.91	1.00
11	1.00	1.00	**1.00**	0.99	0.85	0.95	0.99	**1.00**	0.99	**1.00**
13	0.00	0.00	0.00	0.00	**0.05**	**0.04**	**0.23**	**0.19**	**0.04**	0.02
14	0.00	0.00	0.00	0.00	**0.03**	**0.12**	**0.28**	**0.23**	**0.04**	0.00
16	0.00	0.00	0.24	0.60	0.07	**0.83**	**0.89**	**0.87**	**0.81**	0.24
17	0.00	0.60	**0.63**	0.00	0.16	0.14	0.45	0.34	0.10	0.60
19	0.13	0.00	0.00	**0.97**	0.09	0.11	0.34	0.48	0.02	0.02
20	0.85	1.00	**1.00**	0.00	0.03	0.04	0.34	0.36	0.02	0.93
21	0.02	0.08	0.91	0.98	0.97	0.97	0.97	**0.99**	0.96	0.36
22	1.00	1.00	**1.00**	0.98	0.18	0.01	0.01	0.25	0.09	0.99
23	0.88	0.00	**1.00**	0.71	0.52	0.02	0.02	0.46	0.08	0.99
24	0.00	0.00	0.17	**0.92**	0.31	0.01	0.02	0.65	0.02	0.36
25	0.01	0.00	0.02	0.29	**1.00**	**0.97**	**0.96**	**1.00**	**0.96**	**1.00**
26	0.00	**1.00**	**1.00**	**1.00**	0.99	0.98	0.96	0.99	0.98	0.97
27	0.00	0.20	0.06	0.00	**0.44**	**0.34**	**0.34**	**0.75**	**0.33**	0.01
28	0.94	**1.00**	**1.00**	0.03	0.97	0.96	0.97	0.90	0.96	0.98
29	0.00	0.00	0.00	**0.87**	0.05	0.05	0.02	0.00	0.31	0.00
30	0.00	**1.00**	**1.00**	0.83	0.92	0.96	0.96	0.96	0.96	**1.00**
31	0.00	0.00	0.30	0.79	**0.88**	**0.85**	**0.98**	0.32	**0.99**	0.05
32	0.91	0.00	0.94	-	0.92	**0.94**	**1.00**	**0.94**	**0.94**	0.91
33	0.00	0.98	0.88	-	0.96	0.91	**1.00**	0.89	0.89	0.02
34	0.00	**0.99**	0.60	0.33	0.65	0.02	0.90	0.20	0.04	0.54
35	0.00	0.26	0.00	0.84	0.31	0.02	0.68	0.04	0.01	**0.87**
36	0.12	0.89	0.88	0.81	**0.92**	**0.88**	**0.99**	**0.90**	**0.90**	0.20
37	**1.00**	**1.00**	0.90	0.84	0.79	0.60	0.97	0.79	0.80	0.90
38	0.93	**1.00**	0.86	0.77	0.64	0.07	0.68	0.53	0.04	0.08
39	0.00	0.37	0.91	0.84	0.90	0.85	**0.99**	0.87	0.87	0.87
40	0.93	**1.00**	**1.00**	0.78	0.96	0.84	**1.00**	0.89	0.88	0.85
41	0.36	0.00	0.64	0.00	**0.66**	**0.75**	**0.98**	**0.83**	**0.84**	0.73

5.7.2. Evaluation using SWaT-2019 Dataset

Table 7 presents the evaluation measures for the two proposed approaches with their one-path alternatives for the SWaT-2019 dataset. The IDS-MAN achieved the highest performance for precision, F1-score, accuracy, TNR, and fall behind the best recall by 1%. The IDS-ITS with ENet comes in second place, followed by the IDS-ITS with CC. Generally, the models trained on one modality showed significantly lower performance compared to their counterparts.

Table 7: Comparison Between the IDS-ITS and IDS-MAN for SWaT-2019 Dataset.

Approach		**Precision**	**Recall**	**F1**	**Accuracy**	**TNR**
IDS-ITS	NP	0.242	0.515	0.329	0.573	0.588
	PP	0.208	0.372	0.267	0.584	0.638
	Add	0.248	0.630	0.356	0.538	0.514
	CC	0.441	0.607	0.511	0.764	0.804
	Enet	0.494	0.637	0.557	0.794	0.834
PSA		0.355	**0.804**	0.493	0.684	0.656
NSA		0.210	0.797	0.332	0.388	0.292
IDS-MAN		**0.648**	0.794	**0.714**	**0.878**	**0.898**

To analyze the recall values and understand the detection of the individual attacks that target different parts of the system, Figure 19 shows the individual accuracies of the normal samples, the instances of the exfiltrate historian data attack, the instances of disrupt sensor and actuator attack, and the overall accuracy. The PSA model detected the attacks that disrupt the sensors and actuators with an accuracy of 87% compared to 45% for the network model and 61% for the IDS-MAN. For the attacks that exfiltrate the historian data, the network model obtained a high detection of 98% compared to 77% for the process model and 89% for the IDS-MAN. On the other hand, the IDS-MAN classified 90% of normal instances with an increase of 24% and 61% compared to the PSA and NSA models. The same behavior is observed with the

IDS-ITS, in which the NP model obtained an accuracy of 66% for the attacks that target the historian data while the PP model got an accuracy of 20%. On the contrary, the PP and NP models achieved accuracies of 71% and 21% for the attacks that target the process metrics.

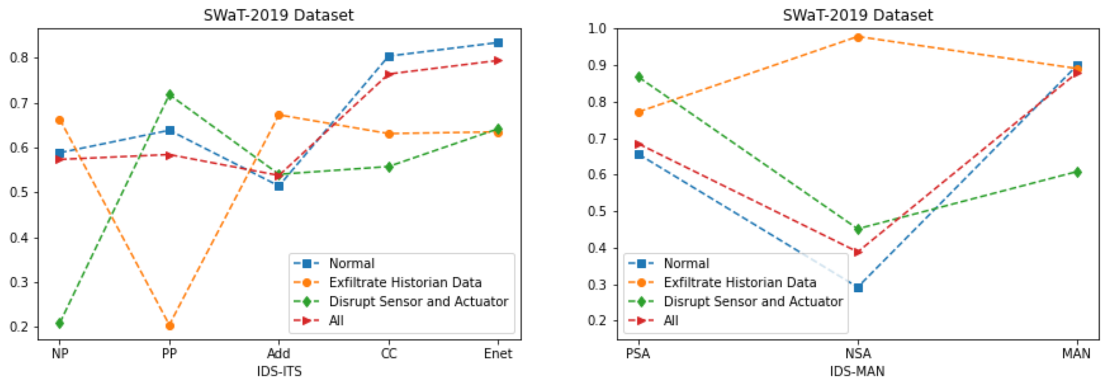

Figure 19: The classification accuracy of the individual sample types for the two approaches and their one-modality alternatives.

5.8. Discussion

5.8.1. IDS Performance

The two proposed IDSs show different performances. As an overall result, the IDS-ITS performed better with the SWaT-2015 dataset in terms of F1-score, recall, and precision. IDS-MAN achieved a higher accuracy; however, comparing accuracy for this scenario is misleading because with IDS-ITS, the data are balanced, while in IDS-MAN, the data are imbalanced. Another important measure to consider when assessing the IDS performance is the TNR (specificity), which is not considered by the previous works. In this measure, the IDS-MAN scored the highest. For IDS, the value of the TNR should not be lower than the TPR (recall) value. Even though the goal of the IDS is to detect cyberattacks and achieve a high TPR, obtaining high TPR with the cost of lower TNR means that the IDS will have a high false-positive rate. The false-positive rate is very critical for IDS in CPS because an action might be triggered to respond to the generated false alert, which may cause unnecessary disruption in the system. Accordingly, both TNR and TPR are equally important for IDS in CPS.

From another perspective, the best-performed IDS-ITS with ENet, which outperforms the IDS-MAN and previous works in terms of F1-score, achieved a lower recall value compared to the IDS-MAN. By comparing these two models, we can observe that the IDS-MAN scored higher in terms of the TNR by 2.6% and TPR by 3.7%. For the individual detection rates of the attacks, the IDS-MAN obtained a higher rate for 19 attacks compared to 16 attacks with the ENet IDS-ITS. Besides, the main advantage of the IDT-ITS is its ability to detect all the attack scenarios, which was not achieved by the IDS-MAN and the previous works. Detecting all the attacks is the top priority for IDSs. Even if the detection rates for some of the attacks are low, however, this will alert the operators, and further actions can be taken based on their assessment of the severity of the attack.

5.8.2. Attacks Discoverability

The intuition behind using different data sources is to increase the discoverability of the attacks and the performance of the IDS by capturing the correlations between the various attributes across the system. For the SWaT-2019 dataset, using data from two different modalities (process-level and network-level) clearly showed that the attacks' discoverability is increased significantly compared to using only one data source. If one data source is used and the attack target is not monitored, some attacks can go undetected by the IDS. For example, for insider attackers with the intent of harming the physical process, a network IDS will have low discoverability for these kinds of attacks since the attack will not leave any footprints in the network traffic characteristics except for the payload. However, including the packets' payload in the IDS introduces the challenge of dealing with unstructured data. Moreover, it does not eliminate the need to consider the process-level data because the packets' payload may not reflect the actual system state. This owes to the fact that the payload can be spoofed to reflect misleading information about the state of the system. Since industrial protocols such as Modbus lack encryption, the attacker can know the target's normal range by observing the network traffic and changing the payload accordingly to reflect the normal operation of the system while causing actual harm to the system.

Besides that, using only IDS at the physical process is also not sufficient since the IDS will not be aware of the cyber level of the system. For example, considering a DOS attack that does not target the physical process directly, the attack most likely will not be detected by process-level IDS until the attack is propagated and causing a major delay in the operation of the process. However, considering the network traffic can help detect these attacks at an early stage and take actions accordingly to avoid major disruptions in the operations of the physical process.

5.8.3. Real-Time Operation

The two proposed approaches have similar requirements for real-time implementation. Both IDS operates in real-time by reading the process-level metrics continuously every specified period (e.g., every 1 second). Then if network traffic is used, the data has to pass through the features extraction stage. After that, the data go through the pre-processing stage, which involves normalizing the coming data using either the mean and variance of the training data or the min and max values. These numbers need to be stored to be able to scale the newly arrived data. Since the window-based approach is used, when the IDSs operate for the first time, they will wait for a period equal to the window size in order to be able to make the first prediction. After that, the windows will start to overlap with the previous windows, and enough history will be available for the IDSs to operate without delay by making a prediction at every slide of the window. For IDS-ITS, another pre-processing step is required, which is converting the data into images. Moreover, if the difference time-series is used, extra processing and memory are needed to calculate the differences between the current readings and their previous values and to scale the data.

For efficiency, CNN is known to be faster than RNN because the RNN operates sequentially and has to keep a memory of the previous hidden states. However, since the input data dimension differs in the two approaches, this observation cannot be generalized. For the IDS-ITS, the image size plays a major role in the efficiency since smaller images size will have a much lower training and testing time. The image size in GAF is controlled by the PAA and is usually determined by the computation capacity or by tuning. Using PAA gives an advantage to the IDS-ITS because it can control the dimension of the input data using a well-established approach and implementation without further consideration. However, a drawback of using PAA is the pre-processing time in which the images constructed using PAA take longer since extra computations are needed.

5.8.4. Recommendation

5.8.4.1. Complexity

The two proposed IDSs have different advantages and disadvantages. First, in terms of complexity, the IDS-ITS requires more preprocessing steps to convert the input time-series to images which will cause a delay in the prediction time. Moreover, the use of 3D CNN increased the computational cost required to train and evaluate the IDS. On the other hand, for the IDS-MAN, the only required preprocessing step is the feature normalization. Also, the IDS uses GRU, which is known to be faster than other RNN alternatives (i.e., LSTM) because it has fewer gates. For the complexity of the network architecture, both CNN and GRU have different hyperparameters. However, the tuning results showed that more layers are needed in the IDS-ITS with the two datasets. Therefore, we can conclude that the IDS-MAN is better in terms of the overall complexity.

5.8.4.2. Generalizability

Any IDS has to be generalizable to be able to apply it to different systems. A common drawback for supervised IDS is that attack scenarios have to exist prior to the deployment and the training of the IDS, which is not always the case in real-time systems. In this work, we take advantage of the availability of datasets that contain different attack scenarios to build our IDS. For the IDS-ITS, the dataset has to be balanced in order for the IDS not to have any bias towards the majority class. This means that to train the network, many attack scenarios have to be presented to the IDS. Typically, for any CPS dataset, the number of anomalies is much lower than the number of normal instances since the occurrences of anomalies are rare. For this reason, we used oversampling and undersampling for balancing the number of instances in the two classes. A major drawback of undersampling of the normal instances is that some of the characteristics of the normal behavior will be lost, which may affect the generalizability of the IDS at the testing time. For the IDS-MAN, the IDS is trained to enhance the generalizability of attack detection by measuring the distance to the class embedding representation rather than using binary classification. The learning of the embedding representation eliminates the need to balance the datasets since the embeddings are learned accumulatively from all the class samples, so there is no bias toward any specific class during training. Moreover, a major advantage of the IDS-MAN is that it can be easily extended for semi-supervised and unsupervised training. Then, rather than performing the detection based on the class with smaller distance or higher similarity, a threshold for the normal behavior will be set using the training data, and the attack detection will be determined if the output from the network exceeds this threshold.

5.8.4.3. Attack Detection

The goal of any IDS is to detect all the attacks while maintaining a low FPR. For the SWaT-2015 dataset, the attack scenarios are not repeated, which means that the IDS is evaluated on attacks that are not seen in the training. This provides an indication that the proposed IDSs have good generalization for detecting unknown cyberattacks. In general, the IDS-ITS provided better attack detection since all the attack scenarios are detected. Moreover, in addition to its superiority in detecting attacks 13 and 14 that were not detected previously, the IDS-ITS showed a significant increase in recall values for some of the attacks compared to the previous works, e.g., attacks 3, 16, and 27. On the other hand, the IDS-MAN obtained lower FPR, but two attacks were not detected. For the SWaT-2019 dataset, both IDS detected the two attack scenarios, with IDS-MAN taking the lead by 15%. Therefore, in terms of attack detection, both IDSs showed superior performance with one of the datasets while obtaining good performance with the other.

5.8.5. Classification Output

In the training of the two proposed IDS, the overall performance was mainly assessed based on the F1-score which provide a balance between the precision and recall. However, in some cases, optimizing a specific evaluation measure is of more interest. For example, since the impacts of cyberattacks on CPS can be very harmful, a high recall value might be more desirable than the overall F1-score to minimize the number of attacks that can go undetected by the IDS. In this case, the IDS has to be trained to favor the correctly detected attacks over other classifications. Since supervised learning is used, the IDS can be trained to give more weight to the attack's samples by providing a higher penalty for their misclassification. Hence the network would focus on reducing the errors for this class. Another solution is to increase the number of samples either by collecting new data or by oversampling. So, by having more attack samples, a slight bias would be introduced towards the attack class, and higher recall value will be obtained. On the other hand, if the distance-based classification in IDS-MAN is replaced by a threshold for the normal behavior, the recall can be increased by decreasing the threshold that determines how far the prediction should be from the normal behavior to be considered as attack.

CHAPTER 6: CONCLUSION AND FUTURE WORK

Protecting CPS from cyberattacks has been a priority due to the catastrophic consequences that such attacks can cause. In this thesis, we report the design, implementation, and evaluation of two intrusion detection systems (IDS) based on deep learning and multimodal techniques to detect cyberattacks on the process and network layers of CPS without the need to have a deep understanding of the underlying process. In the first IDS, named IDS-ITS, we used Gramian Angular Field to convert CPS time series data into images to be used for training a CNN classifier for detecting attacks. The evaluation results of the proposed approach demonstrated better detection accuracy, and more attacks were discovered. The second IDS, named IDS-MAN, used multimodal attention techniques for enhanced learning of joint representation between multiple modalities in CPS by focusing on the most important features in the system. Moreover, we explore different merging techniques to combine the different inputs. The main advantage of using multimodal fusion techniques is to capture the cross-modal correlation. The proposed models were evaluated using two datasets. The first dataset contains only process-level data, while the second contains data from the process and network levels.

In the IDS-ITS, the two multimodal merging techniques, which are Compact Bilinear Pooling (CBP) and EmbraceNet (ENet), achieved significant improvement compared to the simple merge techniques. Similarly, in IDS-MAN, the model showed a performance degradation when the multimodal attention layer is replaced by simple merge techniques. Moreover, the results showed that considering different data sources across the CPS for the IDS can boost the performance and the effectiveness of the attacks' discoverability and the overall IDS accuracy. The findings of the thesis could successfully answer the thesis questions and the stated hypothesis. We were able to answer the first research question by using different multimodal merge techniques to show how these techniques can be adopted for building two completely different IDSs for CPS. For the second research question, the IDS-ITS showed that the multimodal merge techniques provided better detection capability compared to the other techniques. Moreover, both IDSs showed a considerable enhancement in the performance of the IDS that considered different modalities compared to the IDS that only used data collected at one CPS layer.

For future work, we will consider the use of dimensionality reduction techniques to enhance the efficiency as well as the performance of the IDS since some attributes may not be significant for the classification, which may lead to a negative impact on the IDS performance. These techniques are especially needed for the network features since the network traffic data are not structured, and many features can be extracted considering the different protocols and the huge amount of data collected at each second. Moreover, we will apply further hyper-parameters optimization to boost the classifier performance and evaluate IDSs using other datasets and testbeds. Another future research direction is to consider the techniques of imaging time series and multimodal attention for semi-supervised and unsupervised learning by considering other types of neural networks such as autoencoders and generative adversarial networks to enhance the generalizability of the IDS. The IDS-MAN can be extended to semi-supervised learning by clustering the unlabeled samples based on their distance to the embedding representations. Also, it can be modified for unsupervised learning by training the network on the normal samples only and setting a threshold to detect the deviations from the system's normal behavior.

One of the limitations of our work is that the proposed IDSs are not tested with scenarios such as natural process disturbances or stealthy attacks. This limitation is mainly due to the lack of datasets that capture such scenarios. Hence, we were not able to perform such evaluation. For future work, we are considering the construction of a multiclass multimodal dataset that includes different attacks and normal scenarios with data captured from different layers by using a CPS testbed. For the normal scenarios, natural disturbances and malfunction scenarios will be included to have a richer dataset that reflects the real-time operation scenarios in CPS. For the attack scenarios, several attack types that target different layers of the CPS will be implemented. The attacks will include the known attacks such as Denial of Service (DOS) and ARP spoofing. In addition, more specific attacks for the used industrial protocol and the process will be considered, such as change the function code of the Modbus packet or tamper a specific sensor/actuator value in the process. The dataset will include data from the different CPS layers, including the physical process, the network traffic, and the hosts such as the PLCs and HMIs. This will enable enhancing the proposed IDS to distinguish between the abnormal behaviors that could occur due to disturbances in the process or malfunction in the system and the malicious attack scenarios. Also, the IDS can be extended to determine the attack types and the attack targets.

REFERENCES

[1] L. Monostori, “Cyber-Physical Systems,” in *CIRP Encyclopedia of Production Engineering*, The International Academy for Production, S. Chatti, and T. Tolio, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2018, pp. 1–8.

[2] K. A. Stouffer, V. Y. Pillitteri, S. Lightman, M. Abrams, and A. Hahn, “Guide to Industrial Control Systems (ICS) Security,” Jun. 2015. Accessed: Feb. 27, 2021. [Online]. Available: https://www.nist.gov/publications/guide-industrial-control-systems-ics-security.

[3] Z. Drias, A. Serhrouchni, and O. Vogel, “Taxonomy of attacks on industrial control protocols,” in *2015 International Conference on Protocol Engineering (ICPE) and International Conference on New Technologies of Distributed Systems (NTDS)*, Jul. 2015, pp. 1–6, doi: 10.1109/NOTERE.2015.7293513.

[4] S. Zhioua, “The Middle East under Malware Attack Dissecting Cyber Weapons,” in *2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops*, Jul. 2013, pp. 11–16, doi: 10.1109/ICDCSW.2013.30.

[5] M. N. Al-Mhiqani *et al.*, “Cyber-Security Incidents: A Review Cases in Cyber-Physical Systems,” *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 9, no. 1, Art. no. 1, 55/31 2018, doi: 10.14569/IJACSA.2018.090169.

[6] C. Alcaraz and S. Zeadally, “Critical infrastructure protection: Requirements and challenges for the 21st century,” *International Journal of Critical Infrastructure Protection*, vol. 8, pp. 53–66, Jan. 2015, doi: 10.1016/j.ijcip.2014.12.002.

[7] M. H. Monzer, K. Beydoun, and J. FLAUS, “Model based rules generation for Intrusion Detection System for industrial systems *,” in *2019 International*

Conference on Control, Automation and Diagnosis (ICCAD), Jul. 2019, pp. 1–6, doi: 10.1109/ICCAD46983.2019.9037882.

[8] T. H. Morris, B. A. Jones, R. B. Vaughn, and Y. S. Dandass, "Deterministic Intrusion Detection Rules for MODBUS Protocols," in *2013 46th Hawaii International Conference on System Sciences*, Jan. 2013, pp. 1773–1781, doi: 10.1109/HICSS.2013.174.

[9] D. Ramachandram and G. W. Taylor, "Deep Multimodal Learning: A Survey on Recent Advances and Trends," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 96–108, Nov. 2017, doi: 10.1109/MSP.2017.2738401.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017, doi: 10.1145/3065386.

[11] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," *arXiv:1512.03385 [cs]*, Dec. 2015, Accessed: Mar. 30, 2021. [Online]. Available: http://arxiv.org/abs/1512.03385.

[12] K. Cho *et al.*, "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, Qatar, Oct. 2014, pp. 1724–1734, doi: 10.3115/v1/D14-1179.

[13] D. Ghosal, M. S. Akhtar, D. Chauhan, S. Poria, A. Ekbal, and P. Bhattacharyya, "Contextual Inter-modal Attention for Multi-modal Sentiment Analysis," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, Oct. 2018, pp. 3454–3466, doi: 10.18653/v1/D18-1382.

[14] C. Hori *et al.*, “Attention-Based Multimodal Fusion for Video Description,” in *2017 IEEE International Conference on Computer Vision (ICCV),* Oct. 2017, pp. 4203–4212, doi: 10.1109/ICCV.2017.450.

[15] H. Nam, J.-W. Ha, and J. Kim, “Dual Attention Networks for Multimodal Reasoning and Matching,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR),* Honolulu, HI, Jul. 2017, pp. 2156–2164, doi: 10.1109/CVPR.2017.232.

[16] S. Han, M. Xie, H. Chen, and Y. Ling, “Intrusion Detection in Cyber-Physical Systems: Techniques and Challenges,” *IEEE Systems Journal,* vol. 8, no. 4, pp. 1052–1062, Dec. 2014, doi: 10.1109/JSYST.2013.2257594.

[17] X. He, E. Robards, R. Gamble, and M. Papa, “Anomaly Detection Sensors for a Modbus-Based Oil and Gas Well-Monitoring System,” in *2019 2nd International Conference on Data Intelligence and Security (ICDIS),* Jun. 2019, pp. 1–8, doi: 10.1109/ICDIS.2019.00008.

[18] A. Golabi, A. Erradi, A. Tantawy, and K. Shaban, “Detecting False Data Injection Attacks in Linear Parameter Varying Cyber-Physical Systems,” in *2019 International Conference on Cyber Security for Emerging Technologies (CSET),* Oct. 2019, pp. 1–8, doi: 10.1109/CSET.2019.8904913.

[19] S. D. Anton, S. Kanoor, D. Fraunholz, and H. D. Schotten, “Evaluation of Machine Learning-based Anomaly Detection Algorithms on an Industrial Modbus/TCP Data Set,” in *Proceedings of the 13th International Conference on Availability, Reliability and Security,* New York, NY, USA, Aug. 2018, pp. 1–9, doi: 10.1145/3230833.3232818.

[20] M. R. Gauthama Raman, N. Somu, and A. P. Mathur, “Anomaly Detection in Critical Infrastructure Using Probabilistic Neural Network,” in *Communications in*

Computer and Information Science, Nov. 2019, vol. 1116 CCIS, pp. 129–141, doi: 10.1007/978-981-15-0871-4_10.

[21] J. Goh, S. Adepu, M. Tan, and Z. S. Lee, “Anomaly detection in cyber physical systems using recurrent neural networks,” 2017, doi: 10.1109/HASE.2017.36.

[22] M. Kravchik and A. Shabtai, “Detecting cyber attacks in industrial control systems using convolutional neural networks,” Toronto, Canada, 2018, doi: 10.1145/3264888.3264896.

[23] D. Shalyga, P. Filonov, and A. Lavrentyev, “Anomaly Detection for Water Treatment System based on Neural Network with Automatic Architecture Optimization,” 2018, [Online]. Available: http://arxiv.org/abs/1807.07282.

[24] J. Inoue, Y. Yamagata, Y. Chen, C. M. Poskitt, and J. Sun, “Anomaly detection for a water treatment system using unsupervised machine learning,” 2017, doi: 10.1109/ICDMW.2017.149.

[25] H. Dong and D. Peng, “Research on abnormal detection of ModbusTCP/IP protocol based on one-class SVM,” in *2018 33rd Youth Academic Annual Conference of Chinese Association of Automation (YAC)*, May 2018, pp. 398–403, doi: 10.1109/YAC.2018.8406407.

[26] D. Li, D. Chen, B. Jin, L. Shi, J. Goh, and S. K. Ng, “MAD-GAN: Multivariate Anomaly Detection for Time Series Data with Generative Adversarial Networks,” 2019, doi: 10.1007/978-3-030-30490-4_56.

[27] J. Kim, J.-H. Yun, and H. C. Kim, “Anomaly Detection for Industrial Control Systems Using Sequence-to-Sequence Neural Networks,” in *Computer Security*, Cham, 2020, pp. 3–18, doi: 10.1007/978-3-030-42048-2_1.

[28] M. Elnour, N. Meskin, and K. M. Khan, “Hybrid Attack Detection Framework for Industrial Control Systems using 1D-Convolutional Neural Network and

Isolation Forest," in *2020 IEEE Conference on Control Technology and Applications (CCTA),* Aug. 2020, pp. 877–884, doi: 10.1109/CCTA41146.2020.9206394.

[29] Q. Lin, S. Adepu, S. Verwer, and A. Mathur, "TABOR: A Graphical Model-based Approach for Anomaly Detection in Industrial Control Systems," in *Proceedings of the 2018 on Asia Conference on Computer and Communications Security*, New York, NY, USA, May 2018, pp. 525–536, doi: 10.1145/3196494.3196546.

[30] Y. Li, L. Zhang, Z. Lv, and W. Wang, "Detecting Anomalies in Intelligent Vehicle Charging and Station Power Supply Systems With Multi-Head Attention Models," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 1, pp. 555–564, Jan. 2021, doi: 10.1109/TITS.2020.3018259.

[31] A. R. Javed, M. Usman, S. U. Rehman, M. U. Khan, and M. S. Haghighi, "Anomaly Detection in Automated Vehicles Using Multistage Attention-Based Convolutional Neural Network," *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–10, 2020, doi: 10.1109/TITS.2020.3025875.

[32] Z.-Q. Qin, X.-K. Ma, and Y.-J. Wang, "ADSAD: An unsupervised attention-based discrete sequence anomaly detection framework for network security analysis," *Computers & Security*, vol. 99, p. 102070, Dec. 2020, doi: 10.1016/j.cose.2020.102070.

[33] A. Kundu, A. Sahu, E. Serpedin, and K. Davis, "A3D: Attention-based auto-encoder anomaly detector for false data injection attacks," *Electric Power Systems Research*, vol. 189, p. 106795, Dec. 2020, doi: 10.1016/j.epsr.2020.106795.

[34] C. Liu, Y. Liu, Y. Yan, and J. Wang, “An Intrusion Detection Model With Hierarchical Attention Mechanism,” *IEEE Access*, vol. 8, pp. 67542–67554, 2020, doi: 10.1109/ACCESS.2020.2983568.

[35] C. Yang, C. Yang, Z. Chen, and N. Lo, “Multivariate Time Series Data Transformation for Convolutional Neural Network,” in *2019 IEEE/SICE International Symposium on System Integration (SII)*, Jan. 2019, pp. 188–192, doi: 10.1109/SII.2019.8700425.

[36] C.-L. Yang, Z.-X. Chen, and C.-Y. Yang, “Sensor Classification Using Convolutional Neural Network by Encoding Multivariate Time Series as Two-Dimensional Colored Images,” *Sensors*, vol. 20, no. 1, Art. no. 1, Jan. 2020, doi: 10.3390/s20010168.

[37] A. B. Said and A. Erradi, “Deep-Gap: A Deep Learning Framework for Forecasting Crowdsourcing Supply-Demand Gap Based on Imaging Time Series and Residual Learning,” in *2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, Dec. 2019, pp. 279–286, doi: 10.1109/CloudCom.2019.00048.

[38] J.-P. Eckmann, S. O. Kamphorst, and D. Ruelle, “Recurrence Plots of Dynamical Systems,” *EPL*, vol. 4, no. 9, pp. 973–977, Nov. 1987, doi: 10.1209/0295-5075/4/9/004.

[39] H. Lee and B. Ko, “Fund Price Analysis Using Convolutional Neural Networks for Multiple Variables,” *IEEE Access*, vol. 7, pp. 183626–183633, 2019, doi: 10.1109/ACCESS.2019.2959022.

[40] Z. Wang and T. Oates, *Encoding Time Series as Images for Visual Inspection and Classification Using Tiled Convolutional Neural Networks*. 2015.

[41] T. Lin, A. RoyChowdhury, and S. Maji, “Bilinear CNN Models for Fine-Grained Visual Recognition,” in *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015, pp. 1449–1457, doi: 10.1109/ICCV.2015.170.

[42] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell, “Compact Bilinear Pooling,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2016, pp. 317–326, doi: 10.1109/CVPR.2016.41.

[43] A. Fukui, D. H. Park, D. Yang, A. Rohrbach, T. Darrell, and M. Rohrbach, “Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, Austin, Texas, Nov. 2016, pp. 457–468, doi: 10.18653/v1/D16-1044.

[44] J.-H. Choi and J.-S. Lee, “EmbraceNet: A robust deep learning architecture for multimodal classification,” *Information Fusion*, vol. 51, pp. 259–270, Nov. 2019, doi: 10.1016/j.inffus.2019.02.010.

[45] Y. Wen, K. Zhang, Z. Li, and Y. Qiao, “A Discriminative Feature Learning Approach for Deep Face Recognition,” in *Computer Vision – ECCV 2016*, Cham, 2016, pp. 499–515, doi: 10.1007/978-3-319-46478-7_31.

[46] J. Goh, S. Adepu, K. Junejo, and A. Mathur, “A Dataset to Support Research in the Design of Secure Water Treatment Systems,” Oct. 2016.