2022 IEEE 42nd International Conference on Distributed Computing Systems Workshops (ICDCSW) | 978-1-6654-8879-2/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICDCSW56584.2022.00040

DeepIIoT: An Explainable Deep Learning Based Intrusion Detection System for Industrial IOT

Mohammed M. Alani
School of IT Administration and Security
Seneca College of Applied Arts and Technology
Toronto, Canada
m@alani.me

Ernesto Damiani
Center of Cyber-Physical Systems
Khalifa University
Abu Dhabi, United Arab Emirates
ernesto.damiani@ku.ac.ae

Uttam Ghosh
Vanderbilt University
Nashville, USA
uttam.ghosh@vanderbilt.edu

***Abstract*—IoT adoption is becoming widespread in different areas of applications in our daily lives. The increased reliance on IoT devices has made them a worthy target for attackers. With malicious actors targeting water treatment facilities, power grids, and power nuclear reactors, industrial IoT poses a much higher risk in comparison to other IoT application contexts. In this paper, we present a deep-learning based intrusion detection system for industrial IoT. The proposed system was trained and tested using the WUSTL-IIOT-2021 dataset. Testing results showed accuracy exceeding 99% with minimally low false-positive, and false-negative rates. The proposed model was explained using SHAP values.**

***Index Terms*—iiot, intrusion, intrusion detection, deep learning, mlp**

I. Introduction

The adoption of Internet-of-Things (IoT) devices is growing rapidly in different areas of our daily lives. Projections show that the amount of data volume of IoT devices to grow from 13.6 zettabytes in 2019, to 79.4 zettabytes in 2025, according to [1]. This rapid rate of growth comes from the fact that IoT devices are now much cheaper to manufacture, and are supporting a wide range of applications in different contexts, such as smart homes, connected and self-driving vehicles, medical devices, manufacturing and logistics among others. The oil and gas industry also relies on industrial IoT devices that use visual and thermal imaging to detect potential problems in pipelines [2].

Industrial IoT (IIoT) adoption has witnessed an even more rapid growth compared to other areas. Predictions show that the projected market size of IIoT to almost double between 2017, and 2025 [3]. This rapid growth in IIoT usage makes it a highly desirable target for malicious actors.

Although IoT and IIoT devices are similar in terms of the main components, they differ in the following points [4]:

1) Manufacturing: IIoT devices require special hardware specifications to operate in extreme environmental conditions such as very-high or very-low temperatures, under water, or in presence of electromagnetic noise.
2) Scalability: The amount of data generated and handled in IIoT is very large compared to the amount of data generated and handled by home-use IoT devices.
3) Purpose: Most home applications of IoT are based on actuating more than sensing. Hence, most of the data generated by home devices goes unnoticed, or does not require any processing. On the other hand, IIoT devices are more focused on sensing. Hence, they generate high volumes of data that is in most scenarios continuous.

IIoT is currently playing a vital role in the digital transformation in manufacturing known as Industry 4.0 and paving the way to Industry 5.0, . This makes it a target for malicious actors. In terms of risk management, there is a clear difference in the risk generated by a successful attack on a home surveillance camera in comparison to a successful attack on a control unit in a nuclear power plant, or a water sanitation station. The risks posed by vulnerable IIoT devices can be of catastrophic impact if exploited maliciously. For this reason, interest is growing in automatic intrusion detection systems based on Artificial Intelligence and Machine Learning [5]. Most research that address intrusion detection using deep-learning, however, still handles the classifier as a black-box, i.e. it does not provide proper explanation on why the classifier is making a certain decisions.

In this paper, we present an explainable IIoT Intrusion Detection System (IDS) based on deep learning. The proposed system was built as a Multi-Layer Perceptron (MLP) classifier. The classifier is fed features extracted from network-flows to detect different types of attacks, including reconnaissance attacks, backdoors, Denial of Service (DoS), and command injection attacks. The dataset used in training and testing the classifier was WUSTL-IIOT-2021 [6]. This dataset was specifically collected from IIoT devices rather than general IoT devices. The classifier's explanability is supported by the generation of SHapley Additive exPlanations (SHAP) values that provide a good understanding of how the classifier makes its decisions.

The following section will review previous works within the area of our research. Section III will explain the proposed system, and its components. The details of the dataset, along with the pre-processing performed on the dataset are explained in section IV. Section VI explains the details of the experiments conducted along with their results. This section also includes a description of the performance metrics used. The next section discusses the explainability of the model using Shapley values (SHAP). Section VII presents the discussion of the results along with the performance comparison with previous works, while the last section provides the conclusions and future directions in research.

II. Previous Works

Yao et al. presented, in 2019, a hybrid machine-learning-aided intrusion detection system for IIoT [7]. The proposed hybrid system is said to operate in an edge-IIoT scenario divided into a central network part and an edge part. In the proposed architecture, the edge routers are considered master nodes, while the industrial edge part is considered the edge nodes. Due to the limited computing power and resources of edge nodes, light-weight LightGBM algorithm were deployed on them to perform the first step of the intrusion detection. A second step of detection is done on the master node based on the features extracted using the LightGMB algorithm. The paper presented a novel concept. However, it did not present any experimental implementation results relevant to the accuracy of the proposed system.

In 2021, Wu et al. introduced an IIoT intrusion detection system based on Graph Neural Networks (GNN)s in IIoT-enabled smart transportation, smart energy and smart factory contexts [8]. This paper used the older version of the dataset utilized in our research, which is WUSTL-IIOT-2018. The study was only introducing the concept and did not provide any experimental implementation results.

Vulfin et al. presented, in 2021 as well, a paper exploring the use of different machine leanring classifiers to build an IIoT intrusion detection system [9]. The paper utilizes WUSTL-IIOT-2018 dataset as well, and selects six features out of the 41 available in the dataset to use in training and testing. The three types of classifiers tested were random forest, decision trees, and multi-layer perceptron. random forest and multi-layer perceptron gave the highest f1-score of 0.919. However, the paper failed to present any type of validation for the results, and did not compare the results to any similar systems.

Zolanvari et al. proposed, in 2021, ADDAI (Anomaly Detection using Distributed AI) [10]. The proposed system is said to scale to large geographical areas and cover a large number of resources. The proposed system is designed to provide high speed, robustness against a single point of failure, low communication overhead, privacy, and scalability. The proposed system also uses WUSTL-IIOT-2018 dataset for training and testing. Tests showed an average success rate of 98.4% while reducing the communication overhead by half compared with the traditional technique of offloading all the raw sensor data to the cloud.

Kasongo introduced, in 2021, an IIoT intrusion detection system that utilizes Genetic Algorithms (GA) for feature selection, and random forest classifier as its fitness function [11]. After feature selection, several machine learning classifiers were used to detect intrusions such as decision trees, linear regression, and random forest among others. The dataset used for training and testing was UNSW-NB15. Testing results showed that the GA-random-forest combination generated a testing accuracy of 87.61%. Althought the paper claims that the system is built for IIoT, the dataset used is a network intrusion detection dataset that did not include any IoT traffic, nor IoT-specific attacks.

Joel Honsou et al. compared two neural networks-based IDSs for conventional IoT, respectively based on the Learning Quantization Vector and the Radial Basis Function approaches. The latter neural network paradigm does allow, in principle, to extract human-readable rules from the trained network. However these rules, which are similar to fuzzy IF-THEN rules, turn out to differ from each other mostly in the threshold values of conditions, and are therefore difficult to understand for the human security expert [12].

Awotunde et al. published, in 2021, a paper proposing a deep learning based intrusion detection paradigm with a hybrid rule-based feature selection [13]. The proposed system was implemented using a hybrid rule-based feature selection and deep feedforward neural network classifier. The proposed system was tested using NSL-KDD and UNSW-NB15 and delivered an accuracy of 0.99 with a 1.0% false-positive rate for NSL-KDD dataset, and 0.989 accuracy with 1.1% false-positive rate for the UNSW-NB15 dataset. This paper also claims to present an IIoT intrusion detection system while it uses datasets that do not contain any IoT traffic, nor include any IoT-specific attacks

III. DeepIIoT

In this section, we will present an overview of our proposed system; DeepIIoT. DeepIIoT is a deep-learning-based industrial IoT intrusion detection system. It relies on a deep-learning classifier that processes features extracted from network-flow data of captured traffic. Figure 1 shows an overview of the DeepIIoT architecture.

As shown in Figure 1, our architecture is divided into two stages: the development and the deployment stage. In the development stage, the raw data set is examined thoroughly, and a suitable pre-processing plan is built based on this examination. Then, the pre-processed dataset is forwarded to the training stage to produce a trained model that is ready for deployment.

At the deployment stage, a traffic capturing unit captures all incoming and outgoing traffic, and passes it to a network-flow extractor that extracts the required features. The extracted features are then passed to the trained neural model, which performs a prediction based on the data fed into it. This prediction determines if the specific traffic flow is considered malicious or benign. The classifier's inference can be further fed into a host-based firewall that would block the source of the attack. Our pipeline's design is based on the idea of identifying a ground feature set from data, and then to classify the traffic in this feature space. Features are relatively easy to extract from raw network traffic data, as they are usually related to packet headers. Also, feature extraction decouples the data points in the feature space that will be considered by the model from the specifics of raw network data, which may differ from one network scenario to another, e.g. due to different protocol versions. .

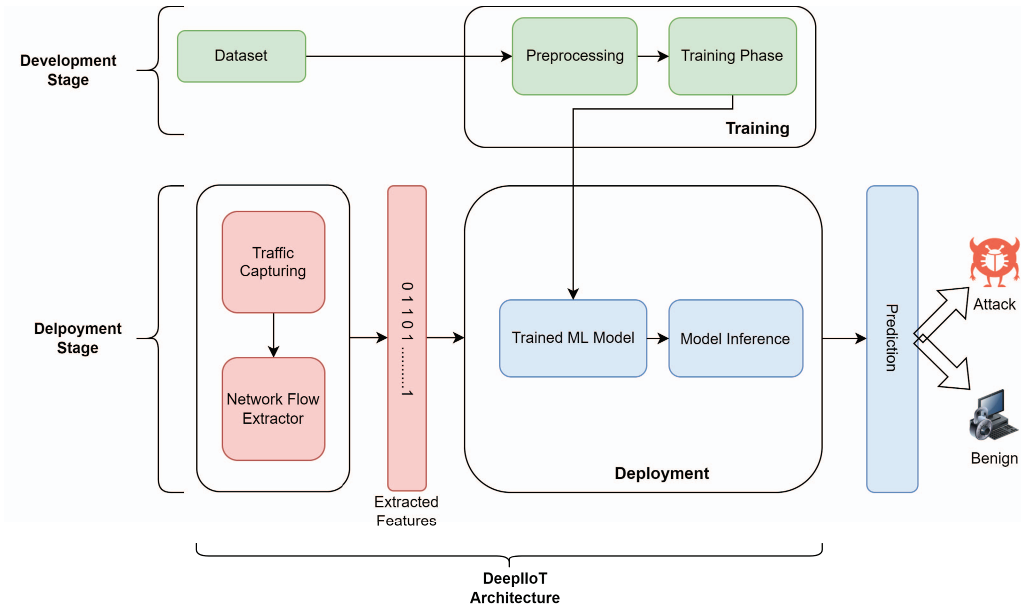

Fig. 1. Overview of DeepIIoT Architecture

IV. The Dataset

The dataset utilized in our experiments is WUSTL-IIOT-2021 [6], which is an updated version of the classic WUSTL-IIOT-2018 dataset. The dataset is composed of 1,194,464 instances with 48 features. Each instance represents one network traffic flow. All features extracted represent information relevant to the network flow such as source IP address, destination IP address, number of packets,etc. Each instance is labelled either malicious, or benign. In addition, each instance has an additional label that represents the specific type of attack. This dataset was chosen because it is specifically focusing on IIoT events and attacks. Specifically, the dataset captures the types of attacks listed below.

1) Command injection attacks
2) DoS attacks
3) Reconnaissance attacks
4) Backdoors

Upon detailed examination of the dataset, we recorded the following observations:

- There is a large imbalance between the benign and malicious classes, as benign flows represent over 92% of the dataset, while malicious flows represent slightly less the 8%.
- There is noticeable imbalance between the different subclasses of attacks. DoS attacks represent about 90% of the malicious traffic, while the other three attack types represent about 10% combined.
- The dataset contains several host-specific features such as the source and destination IP addresses. If used in the training, these host-specific features would could cause severe overfitting. This would mean that the classifier would not generalize well beyond its training dataset.
- The dataset contains some irrelevant features such as the flow start time, and end time.

Based on the observations listed above, we built our dataset pre-processing plan explained in details in the following subsection.

A. Dataset Preprocessing

To address the observations mentioned in section IV, we developed a pre-processing plan including the following steps:

1) The irrelevant features, as well as the host specific features were removed. this includes 'StartTime', 'LastTime', 'SrcAddr', 'DstAddr', 'sIpId', and 'dIpId'. This left the total number of features at 42.
2) As our research goal is to build a high-accuracy intrusion detection system that captures different types of attacks, without categorizing them, we decided to remove the feature that identifies the specific attack type, and rely on the "malicious", and "benign" labels. This left the number of features at 41.
3) To address the class imbalance problem, we performed random under sampling on the benign class to lower the percentage of benign instances from 92% to 66%. This left the dataset with 261,048 instance our of which 174,032 benign, and 87,016 malicious.

At the end of the pre-processing steps, the dataset now has 261,048 instances with 41 features.

V. Experiments and Results

A. Implementation Environment

All experiments were implemented on a computer with the following specifications:

- AMD Ryzen5 3600 4.2GHz computer with 128GB RAM

- GeForce 3060Ti GPU
- Windows 10 Pro operating system
- Python 3.7.11, with Tensorflow 2.3.0, Keras 2.3.1, SciKit-Learn 1.0.1, SHAP 0.39.0, imblearn 0.9.0

B. Performance Metrics

The essential performance metrics of a binary classifier are True Positive(TP), True Negative (TN), False Positive (FP), and False Negative (FN). These four measures, when combined together, generate the confusion matrix.

In our research, we measured the classic four performance metrics that are based on the TP, TN, FP, and FN measures, namely:

1) Accuracy: This metric measures the ratio of correct predictions using the equation:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

2) Precision: This metric measures the ratio of the accuracy of positive predictions using the equation:

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

3) Recall: This metric measures the ratio of positive instances that are correctly detected by the classifier using the equation:

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

4) f1-score: This metric measures the harmonic mean of precision and recall using the equation:

$$f1 - score = 2 * \frac{\frac{TP}{TP+FN} * \frac{TP}{TP+FP}}{\frac{TP}{TP+FN} + \frac{TP}{TP+FP}} \quad (4)$$

C. Experiment Design

The experiments conducted were split into four stages; training, testing, cross-validation, and explainability.

In the first stage, the dataset was randomly split into 75% training subset, and 25% testing subset. The 75% subset was used in training the deep learning classifier. The classifier was trained with the following hyper-parameters:

1) The architecture used was multi-layer perceptron with one input layer, two hidden layers, and an output layer with the number of neurons 41, 64, 12, 1 respectively. The activation function used in the hidden layers was ReLu, and sigmoid for the output layer.
2) The loss function used was binary-crossentropy as the output of the classifier was binary.
3) The metric used for training was ”accuracy”
4) The optimizer used in training was ”adam”.

All experiments at this stage were run for 75 epochs, with a batch-size of 1,000.

Testing was conducted using the 25% of the dataset that was not used in training to ensure classifier's capability to generalize.

Metric	Value
Accuracy	0.9994
Precision	0.9992
Recall	0.9995
f1-score	0.9994
Training Time	25.4487s

TABLE I
TESTING RESULTS FOR THE DEEP NEURAL NETWORK

Fold	Accuracy	Precision	Recall	f1-score
Mean	0.999797	0.999820	0.999723	0.999772
Std Dev	0.000111	0.000116	0.000160	0.000125

TABLE II
10-FOLD CROSS-VALIDATION RESULTS

As a further step to ensure generalization, the third stage of the experiments included 10-fold cross-validation to ensure that the mopdel generalizes well beyond its training dataset.

The last step of the experiment was to run the classifier through SHAP analysis to produce SHAP values for each feature. This information will help in the model's explainabaility as described in section VI.

D. Results

As we tried different architectures of deep neural networks in our initial experiments, we noticed that the performance was poor and the accuracy did not exceed 0.75. To improve the classifier's performance, we normalized the dataset using MinMaxScaler; a normalizer that would rescale the values of each feature independently. When training a model based on network data, feature normalization is crucial to center and normalise the data by subtracting the mean and dividing by the variance. It is important to remark that at each training round the feature normalisation is performed over the data used for training (rather than on the entire dataset) to avoid exposing the model to spurious information on future values (the mean and variance of the whole dataset). Then, we perform normalisation on testing instances as well, but this time using the mean and variance of training variables. In this way, we can evaluate whether our model can generalize well to new, unseen data points. This procedure improved the performance of the classifier dramatically as shown later.

The testing results of the classifier are shown in table I. These performance metrics are calculated as macro average from both classes.

As shown in table I, the classifier is performing very well with an f1-score of 0.9994. Figure 2 shows the confusion matrix plot for the deep neural network test.

As shown in figure 2, the classifier's FP rate is 0.069% only, while the FN rate was 0.032%. Both of these rates are considered very low in comparison to any other IIoT IDS system.

The next step in the experiment was to perform 10-fold cross validation. Table II shows the results of the 10-fold cross-validation process.

As shown in table II, the mean value of all of the metrics is very high. With mean accuracy of 0.9997, and a standard

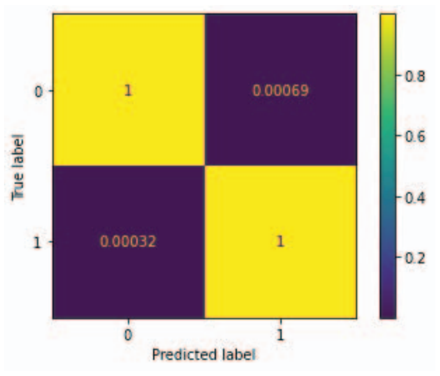

Fig. 2. Confusion matrix plot for the deep neural network

deviation of only 0.0001, these results show that the classifier generalizes well beyond its training dataset. It is also noticeable that the standard deviation of all of the four measured metrics did not exceed 0.0001. This indicates the classifier's stability and robustness, and proves good generalization.

VI. Model Explainability

To increase establish higher trust in our proposed solution, we will work on explaining the trained model's predictions. This practice does not only build trust in the model, but also ensures that the achieved high accuracy originates from explainable conditions, and not resulting from a black-box operation.

In explaining our proposed model, we will employ SHapley Additive exPlanation. SHAP was first introduced in [14] in 2017 as a model-agnostic method to explain machine-learning models that is based on Shapley values taken from game theory. The process involves calculating the impact of each feature by calculating the difference between the model's performance with the feature, and without the feature. This helps in understanding the contribution of the feature to the prediction. The explainer type used in our experiment was DeepExplainer.

Figure 3 shows the SHAP values summary plot of the top ten features with the highest importance. These ten features are ordered in descending order from the feature with the highest impact on the decision to the lowest.

In figure 3, the values shown on the left side of the axis are the values that drag the prediction value down, which makes the prediction closer to "benign", while the values on the right side of the axis pushes the prediction value up which makes the prediction closer to "malware". The dots in red represent a high value of the feature, while the blue dots represent a low value of the feature. For binary features, red dots represent 1, and blue dots represent 0.

As shown in figure 3, the feature with the highest impact is dTtl, which represents the Time-to-Live (TTL) value within the IP packets sent from the destination to the source. The figure shows that the higher value of this feature brings the prediction closer to a lower value, i.e., benign, while the lower value in the feature brings the prediction closer to malicious. This is caused by the fact that the destination (i.e. the target of the attack) sends normal traffic with proper values of TTL when it is not being attacked, while the process that responds to the attack would generate abnormal traffic that is usually crafted by the attacker.

The feature with the second highest impact on the model's prediction is SrcRate. This feature represents the number of packets received from the source(i.e. the attacker)per second. The figure shows that the higher the rate was, the more the prediction is pushed towards a "malicious" prediction. This is consistent with the fact that most attackers send a large influx of messages for the purpose of scanning, brute-forcing, or conducting DoS attacks.

The remaining eight features within the top-10 list can be divided into two types; features that push the prediction to benign when they have a high value (Load, SrcLoad, sTtl, SPort), and features that push the prediction to malicious when they have higher value (Rate, pLoss, DPort, Min). More details on the meaning of these features can be found in [6].

VII. Discussion

Testing, and cross-validation, results showed the DeepIIoT is a reliable intrusion detection system that generalizes well beyond its training dataset. We used the performance measures mentioned in section V-D to build table III that shows a comparison of our proposed system with previous works addressing the same problem.

As shown in table III, our proposed system outperforms other systems in terms of performance. In addition, our system was the only one handling the imbalance issue in the dataset. This imbalance can be clearly seen in the results of [9] that has an accuracy of 0.999 while having an f1-score of 0.919. This clearly indicates that the minority class is being falsely classified as the majority.

The explainability of the model, as described in section VI, aligns well with our expectations of the impact of the features on the classifier's predictions.

VIII. Conclusions and Future Work

In this paper, we presented DeepIIoT; an explainable deep learning-based intrusion detection system for IIoT. The proposed system was trained and tested using WUSTL-IIOT-2021 dataset, and achieved accuracy exceeding 99% with false-positive rate of 0.069%, and a false-negative rate of 0.032%. With these performance measures, DeepIIoT outperformed other research addressing the same problem.

Future directions in our research will deal with the trend toward feature set standardization and its impact on our pipeline's accuracy. Preliminary results by other researchers [17] indicate that adopting a specific set of standardized features for network traffic will improve models' portability across network scenarios and enable transfer learning by domain adaptation. On the other hand, limiting the feature space *a priori* may have an impact on accuracy, perhaps affecting the accuracy of multi-class classification more than the one of

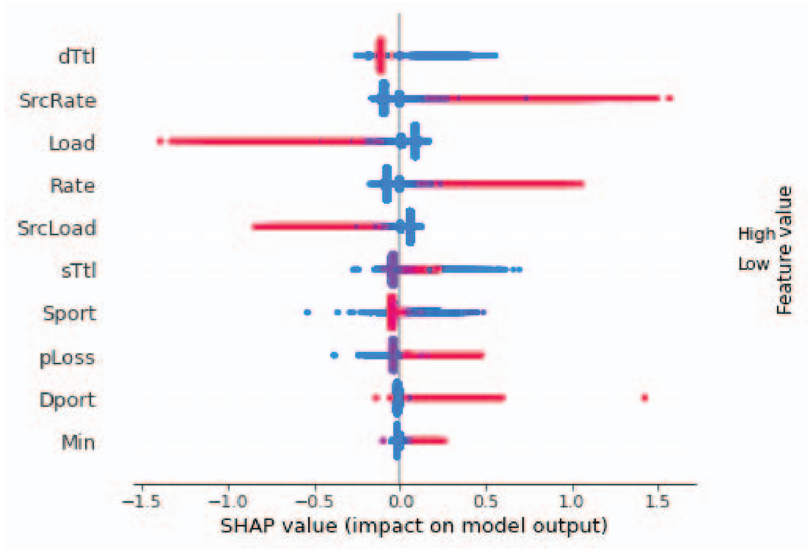

Fig. 3. SHAP Values Summary Plot for the Highest-Impact 10 Features

Ref	Dataset	Features	Classifier	Accuracy	f1-score
[9]	WUSTL-IIOT-2018	6	MLP	0.999	0.919
[15]	WUSTL-IIOT-2018	19	RF	0.9989	-
[16]	[16]	-	Extratrees RF K-NN	0.99 0.99 0.91	0.97 0.97 0.89
DeepIIoT	WUSTL-IIOT-2021	41	MLP	0.9994	0.9994

TABLE III
COMPARISON OF THE PERFORMANCE OF DEEPIIOT TO PREVIOUS WORKS

binary (malicious vs. benign) classifiers. We plan to explore the link between the parameters of the IDS classification problem and the strategy to be adopted for feature extraction in a future paper.

REFERENCES

[1] "Global IoT connections data volume 2019 and 2025 | Statista," Feb 2022. [Online; accessed 23. Feb. 2022], https://www.statista.com/statistics/1017863/worldwide-iot-connected-devices-data-size.

[2] O. E. Ijiga, R. Malekian, and U. A. Chude-Okonkwo, "Enabling emergent configurations in the industrial internet of things for oil and gas explorations: A survey," *Electronics*, vol. 9, no. 8, p. 1306, 2020.

[3] "Global industrial Internet of Things market size 2017-2025 | Statista," Feb 2022. [Online; accessed 23. Feb. 2022], https://www.statista.com/statistics/611004/global-industrial-internet-of-things-market-size.

[4] A. Sari, A. Lekidis, and I. Butun, "Industrial networks and iiot: Now and future trends," in *Industrial IoT*, pp. 3–55, Springer, 2020.

[5] B. Li, Y. Wu, J. Song, R. Lu, T. Li, and L. Zhao, "Deepfed: Federated deep learning for intrusion detection in industrial cyber–physical systems," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5615–5624, 2020.

[6] M. Zolanvari, "Wustl-iiot-2021," 2021. https://dx.doi.org/10.21227/yftq-n229.

[7] H. Yao, P. Gao, P. Zhang, J. Wang, C. Jiang, and L. Lu, "Hybrid intrusion detection system for edge-based iiot relying on machine-learning-aided detection," *IEEE Network*, vol. 33, no. 5, pp. 75–81, 2019.

[8] Y. Wu, H.-N. Dai, and H. Tang, "Graph neural networks for anomaly detection in industrial internet of things," *IEEE Internet of Things Journal*, 2021.

[9] A. Vulfin, V. Vasilyev, S. Kuharev, E. Homutov, and A. Kirillova, "Algorithms for detecting network attacks in an enterprise industrial network based on data mining algorithms," in *Journal of Physics: Conference Series*, vol. 2001, p. 012004, IOP Publishing, 2021.

[10] M. Zolanvari, A. Ghubaish, and R. Jain, "Addai: Anomaly detection using distributed ai," in *2021 IEEE International Conference on Networking, Sensing and Control (ICNSC)*, vol. 1, pp. 1–6, IEEE, 2021.

[11] S. M. Kasongo, "An advanced intrusion detection system for iiot based on ga and tree based algorithms," *IEEE Access*, vol. 9, pp. 113199–113212, 2021.

[12] J. T. Hounsou, P. B. C. Niyomukiza, T. Nsabimana, G. Vlavonou, F. Frati, and E. Damiani, "Learning vector quantization and radial basis function performance comparison based intrusion detection system," in *Intelligent Human Systems Integration 2021* (D. Russo, T. Ahram, W. Karwowski, G. Di Bucchianico, and R. Taiar, eds.), (Cham), pp. 561–572, Springer International Publishing, 2021.

[13] J. B. Awotunde, C. Chakraborty, and A. E. Adeniyi, "Intrusion detection in industrial internet of things network-based on deep learning model with rule-based feature selection," *Wireless communications and mobile computing*, vol. 2021, 2021.

[14] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," *Advances in neural information processing systems*, vol. 30, 2017.

[15] M. A. Teixeira, M. Zolanvari, K. M. Khan, R. Jain, and N. Meskin, "Flow-based intrusion detection algorithm for supervisory control and data acquisition systems: A real-time approach," *IET Cyber-Physical Systems: Theory & Applications*, 2021.

[16] Y. Song, W. Luo, J. Li, P. Xu, and J. Wei, "Sdn-based industrial internet security gateway," in *2021 International Conference on Security, Pattern Analysis, and Cybernetics (SPAC)*, pp. 238–243, IEEE, 2021.

[17] M. Sarhan, S. Layeghy, N. Moustafa, and M. Portmann, "Netflow datasets for machine learning-based network intrusion detection systems," in *Big Data Technologies and Applications*, pp. 117–135, Springer, 2020.