International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Real-Time Intrusion Detection For IIOT: Advancing Edge Computing Security with Machine Learning-Based Solutions

[1]Afnan Ziyad Alrubayyi, [2]A. A. Abd El-Aziz, [3]Osama Ouda

Submitted: 03/02/2024 **Revised**: 11/03/2024 **Accepted**: 17/03/2024

Abstract: The emergence of IIoT in vital industrial systems has heightened the demand for strong security frameworks as cyber threats become more sophisticated. However, the traditional security measures are usually inadequate, especially in real-time operational scenarios, making the industrial systems susceptible to disruptive attacks. This is the problem that this paper tries to solve by proposing a new Intrusion Detection System (IDS) for IIoT that uses Machine Learning (ML) and Deep Learning (DL) approaches, supplemented by Neighborhood Component Analysis (NCA) for improved feature selection. Our holistic method is based on pre-processing the dataset, which includes one-hot encoding and min-max normalization, applying NCA for the most critical features selection, and using a two-level classification strategy, where ensemble classifiers, standard classifiers, and deep neural networks are used. IIoT based IDS shows enhanced performance with the use of NCA in feature selection. NCA-optimized models deliver near perfect values of accuracy, precision, recall, F1-score, and AUC, which are close to 1, outperforming Without-NCA. Moreover, the NCA usage decreases both training and testing times significantly, making the system much more effective for real-time applications.

Keywords: *Industrial Internet of Things (IIoT), Intrusion Detection System (IDS), Machine Learning (ML), Deep Learning (DL), and Neighborhood Component Analysis (NCA).*

I. INTRODUCTION

The concept of the Internet of Things (IoT) has revolutionized the digital age by making physical objects become part of computer systems, improving the efficiency, economic advantages, and reducing human input. At the heart of IoT is the interconnectivity of ordinary devices that are connected to the internet, and thus, they can send and receive data indefinitely. These technological changes resulted in the emergence of smart houses, healthcare monitoring systems, and automatic industrial operations. With the development of IoT technologies, the complexity of the generated data and the amount of the data have increased dramatically, which creates new problems for data management and system security [1].

IIoT expands the scope of these technologies to key industrial sectors like manufacturing, logistics, oil and gas, and agriculture. Utilizing machine learning (ML) and big data technologies, IIoT is to make the industrial operations more reliable, efficient, and safe via improved sensor data analysis, machine-to-machine (M2M) communication, and advanced automation processes. The stakes are highest in industrial environments, where system failures can result in huge economic losses or disasters, emphasizing the importance of robust and reliable IIoT systems [1].

Constrained by the shortcomings of conventional centralized computing modes, edge computing has emerged as a critical innovation by processing data much closer to where the data is produced. This change decreases the latency to a large extent, saves network bandwidth and improves system responsiveness, which are critical requirements in IIoT environments as quick, real-time decisions are the key to keeping the operational integrity and safety. Nevertheless, the use of edge computing also brings new security threats. Edge devices are distributed and subject to both physical and cyber threats, which makes the management of the huge data quantities created and their integrity in transit or at rest a challenge [2, 3].

The conventional Intrusion Detection Systems (IDS) comprising network-based (NIDS) and host-based (HIDS) systems have played a critical role in keeping the security threat under control. NIDS monitors the network traffic to find anomalies, while HIDS looks at system logs for malicious activities. The adoption of machine learning techniques with IDS technologies is a breakthrough as it enhances their ability to detect advanced and dynamic threats more quickly. The machine learning-based IDS systems are dynamic in nature, thus they are becoming more accurate and eliminating the false positives that improves the security edge computing in the IIoT networks [4].

Based on these developments, this paper proposes an advanced IDS for IIoT that employs more sophisticated

[1]*Master student of Information Systems College of Computer and Information Sciences, Jouf University Sakaka 72388, Saudi Arabia 441205043@ju.edu.sa*
[2]*Department of Information Systems, College of Computer and Information Sciences, Jouf University, Sakaka 72388,SaudiArabia; aaeldamarany@ju.edu.sa Department of Information Systems and Technology, Faculty of Graduates Studies and Research, Cairo University, Giza 12613, Egypt Faculty of Graduate Studies for Statistical Research*
[3]*Department of Computer Science, College of Computer and Information Sciences, Jouf University, Sakaka 72388, Saudi Arabia*

machine learning (ML), and deep learning (DL) methods. Our system focuses on the integration of the benefits of both ML and DL models for the purpose of detecting and response to security threats efficiently. The principal innovation in our approach is the incorporation of Neighborhood Component Analysis (NCA) for feature selection which it reduces the dimensionality of the dataset significantly and hence the computational load of the IDS that leads to better performance. This helps our system to rapidly identify usual behaviors as well as potential security threats of different degrees of magnitude. The work will define the architecture of the envisioned IDS, the utilized method, and the anticipated consequences of our developments to enhance the security and operational performance of the IIoT networks.

II. RELATED WORK

In the section, various recent studies relevant to our work are summarized and presented in Table 1 as follows:

Yao et al. [5] offer a novel approach for intrusion detection in the IIoT based on the union of machine learning and edge computing. This hybrid system has been developed to boost the efficiency of the IIoT environments by processing data at the edge of the network. With data analysis being localized, the system effectively eliminates the latency problems usually associated with central processing systems. The ability to instantly process the phenomenon is very crucial in the maintenance of the ability to react timely to potential security threats that can be important in preventing operational interruptions or data leaks in industrial environments. In addition, the system employs machine learning algorithms that serve to improve the system's performance in terms of threat detection concerning current data. Such a learning process helps in the effective identification of known and novel security risks and thus improves the security architecture of IIoT networks. The centrality of data processing is reduced leading to reductions in network traffic that in turn makes the system more efficient by using less bandwidth and by reducing the load on central servers.

Eid et al. [6] introduces machine learning techniques to be used for an intrusion detection system in Industrial Internet of Things (IIoT) networks. This study is significant as it investigates the potential of utilizing machine learning to improve IIoT security systems, which are currently at the heart of cybercriminals' attacks because of their critical role in industrial operation. The research assesses machine learning models to identify those that are more appropriate for identification of abnormal and potentially dangerous activities in IIoT networks. Responsive models are designed to be reactive and reliable in real-time, identifying and responding to threats within time, hence preventing harm.

The paper of Zolanvari et al. [7] concerned with the analysis of network vulnerabilities in the industrial Internet of Things (IIoT) utilizing the machine learning approaches. This study is essential because it addresses the urgent requirement for robust security solutions in IIoT networks that are the fundamental part of the contemporary industrial systems that are exposed to the growing number of sophisticated cyber-attacks. The paper evaluates the performance of different machine learning models in the identification and analysis of vulnerabilities of IIoT networks by the authors. The goal is to achieve a thorough understanding of all possible security failures prior to their exploitation by hackers. The dynamic security protocol is the cornerstone of the integrity and quality of industrial activities where interlinked devices are continuously working. The results of this study are a part of the bigger field of cybersecurity: they provide a vision of how machine learning can be used to enhance security architectures of IIoT systems, that will make industrial environments safer.

Guezzaz et al. [8] propose a light weighted hybrid edge-based IIoTedge basedintrusion detection framework. This study explores the growing demand for best-of-breed security systems that work in an edge computing environment - an issue that is crucial to Industrial Internet of Things (IIoT). The approach outlined is to combine edge computing speed and local processing advantage with machine learning analytical power to detect and respond to security threats on the fly. The hybrid nature of the framework makes it lightweight, i. e., it does not require many computational resources, which is vital to maintain the performance of edge devices, which are usually low in processing power. The findings of the study hold potential in improving the security of IoT systems by using machine learning for quick detection and reaction to potential threats, thus protecting the critical industrial processes.

Ferrag et al. [9] introduce "Edge-IIoTset," a new and comprehensive cybersecurity dataset specially designed for IoT and IIoT applications. This dataset was built keeping in mind the requirement for creating and testing machine-learning models that are used in centralized as well as in federated learning settings, which is an essential characteristic in enhancing the effectiveness and the response of cybersecurity measures. The significance of "Edge-IIoTset" lies in its real-world practicality and detail which offers researchers and developers many practical data cases covering the complexity and diversity of modern IoT and IIoT environments. The authors' suggested dataset will include several attack scenarios as well as normal data that will enable the formulation of machine learning algorithms that will be able to detect, analyze, and respond to the threat in cyber security. This resource is especially useful in building intrusion detection systems and other security solutions which need intensive training data. This kind of dataset access will lead to better benchmarking and IoT and IIoT infrastructure specific security solutions development.

Ullah et al. dedicated solution for an IIoT network is MARGU-IDS, which is a high-efficient intrusion detection system [10]. This system introduces a Multi-Head Attention-Based Gated Recurrent Unit (MAGRU) method, an advanced method which combines attention mechanisms with gated recurrent units (GRUs) for better detection of malicious activities in IIoT environments. Integration of multi-head attention mechanism into MAGRU-IDS allows it to process multiple fragments of input data sequences simultaneously, which enhances its efficiency in the detection of intrusion patterns. This is especially advantageous in IIoT environments because the data flows are continuous and complicated and therefore need advanced analytics that can adjust to the dynamic changes in the performance of the network. The GRUs can carry out the processing of these data sequences efficiently where the previous inputs' information is retained without the vanishing gradient issue that occurs in the normal recurrent neural networks.

Vaiyapuri et. al [11] examine the potential of deep learning technologies in enhancing intrusion detection in Industrial Internet of Things (IIoT) networks. This study critically evaluates various deep learning approaches and their efficiencies and outlines the areas for future research in terms of the security of IIoT systems. The authors provide details on

Table 1: Summary of paper presented in related work.

Authors	Year	Key Findings	Limitations
Yao et al. [5]	2019	Developed a hybrid IDS that combines machine learning with edge computing to reduce latency and network traffic in IIoT systems.	Limited real-time processing capabilities.
Eid et al. [6]	2023	Explored the effectiveness of various ML models in detecting unusual activities within IIoT networks.	Lack of detail on the specific computational requirements and scalability of the ML models used.
Zolanvari et al. [7]	2019	Assessed the effectiveness of ML models in identifying vulnerabilities within IIoT networks.	Primarily theoretical; lacks extensive real-world testing.
Guezzaz et al. [8]	2022	Introduced a lightweight hybrid IDS framework using ML for edge based IIoT security.	Framework might not scale well in highly dynamic or expansive network environments.
Ferrag et al. [9]	2022	Presented "Edge-IIoTset," a realistic cybersecurity dataset for IoT and IIoT applications.	Actual effectiveness of the dataset in diverse real-world scenarios remains to be fully tested.
Ullah et al. [10]	2023	Developed MAGRU-IDS using a multi-head attention-based GRU for enhanced intrusion detection in IIoT.	Complexity of the model might impact real-time deployment in resource-constrained environments.
Vaiyapuri et al. [11]	2021	Reviewed deep learning approaches for intrusion detection in IIoT networks, outlining opportunities and future directions.	Detailed empirical research to validate these models in operational settings is still needed.
Tang et al. [12]	2023	Investigated the use of deep learning algorithms to improve IDS performance in complex network environments.	Challenges in deployment and integration with existing security infrastructures were not addressed.
Rodríguez et al. [13]	2023	Implemented an attentive transformer deep learning model with XAI for intrusion detection in IoT systems.	Exploration of practical implications and deployment of XAI in operational environments is limited.
Saxena et al. [14]	2022	Provided a comprehensive review of deep learning implementations in IIoT, highlighting potential and challenges.	Lacks detailed case studies and in-depth analysis of real-world application successes and failures.

the advantages of deep learning in detecting subtle patterns and anomalies that might be missed outright by traditional methods, making it suitable for the dynamic and heterogeneous nature of IIoT. The paper talks about several deep learning models that have shown good performance in the detection of intrusive activities from network traffic data; these models include Convolutional Neural networks (CNNs), Recurrent Neural Networks (RNNs), and Autoencoders. In addition, this paper features the limitations of deep learning models, which include large datasets and computational requirements and research issues such as practical training methods and models that can work well with fewer resources.

Tang et al. [12] are focused on the improvement of Intrusion Detection Systems (IDS) with deep learning algorithms. This study is part of a broader field of cybersecurity technologies that are covered at the same time in other conferences such as Pervasive Intelligence and Computing, Cloud and Big Data Computing, and Cyber Science and Technology Congress. The authors carried out research on the capability of deep learning to greatly enhance the accuracy and efficiency of intrusion detection in complicated networks. They also discuss a variety of deep learning models that can understand and make intelligent analyses of huge network datasets without creating any delay and also find anomalies faster and potentially malicious acts. Built on the power of deep learning which lets the system learn from the new data and adapt without explicit programming, the system is aimed at being an efficient solution to the perpetually changing cyber threats world.

Rodríguez et. et al [13] introduce an innovative deep learning method for intrusion detection in IoT systems, that relies on an attention transformer model with automated interpretable feature selection. This study is significant because of the union of cutting-edge machine learning methods and explainability, which is a new requirement in AI and cybersecurity. The use of attentive transformers in their model allows for a concentrated and detailed processing of sequential data, which is typical of network traffic. This approach allows the model to focus on the most relevant parts of the data necessary for identifying potential threats, thereby enhancing the performance of the intrusion detection system. Furthermore, the use of explainable AI (XAI) techniques makes the system not just predict and detect intrusions, but also reveal ‘how’ and ‘why’ decisions are taken. In such systems, transparency is crucial for trust and accountability, especially in IoT environments. The approach of the study in blending deep learning with XAI creates the basis for the creation of more powerful and understandable security solutions in the IoT environment. The development is important, since it reduces the threat of emergence of more complex cyber dangers.

Saxena et. al [14] provide an analysis of how deep learning technologies are utilized in the Industrial Internet of Things (IIoT). Their study emphasizes the transformative power of deep learning in improving the IIoT system’s functions, from predictive maintenance to real-time data analysis and further. The writers dwell on both the pros and cons of the application of deep learning methods in IIoT. They argue that in spite of the fact that deep learning can significantly improve operational efficiencies and insights through advanced data analysis and pattern recognition, there are such problems as the necessity to have large data sets, high computational power, and concerns about data privacy and security. This paper is a treasure trove of information on deep learning in the context of IIoT of today which outlines all the latest developments and identifies several key areas in which more research and development is necessary.

Our work seeks to address several key gaps in the field of Intrusion Detection Systems for the Industrial Internet of Things by leveraging advanced machine learning and deep learning techniques:

- **Comparative Analysis of ML and DL Models**: We are exploring how different ML models stack up against DL models in terms of effectiveness in detecting and responding to security threats within IIoT environments.
- **Efficient Data Processing with NCA:** By implementing NCA for feature selection, our research addresses the need for more efficient data processing methods. This approach significantly reduces the dimensionality of data, enhancing the IDS's performance and lowering the computational demands, which is crucial for real-time applications in IIoT.
- **Multi-level Threat Detection:** Unlike many existing systems that only categorize activities as normal or malicious, our IDS is designed to identify various levels of threats, providing a nuanced and effective security mechanism.

Our work contributes to enhancing the security frameworks of IIoT systems, offering both theoretical insights and practical solutions to better protect industrial environments against sophisticated cyber threats.

III. METHODOLOGY

The proposed system framework comprises three primary components: the recorder, an AI-based intrusion detection system placed on the edge layer server, and the decision-making component. The recorder is responsible for capturing and recording the behavior of IoT sensors and networks. Simultaneously, the AI-based system utilizes the properties of the IoT network to analyze and detect its status. The information about the IoT network's status is then transmitted to the decision-making component, which, based on the network behavior, determines whether it is operating normally

or experiencing an attack. Figure 1 visually depicts the components of this framework.

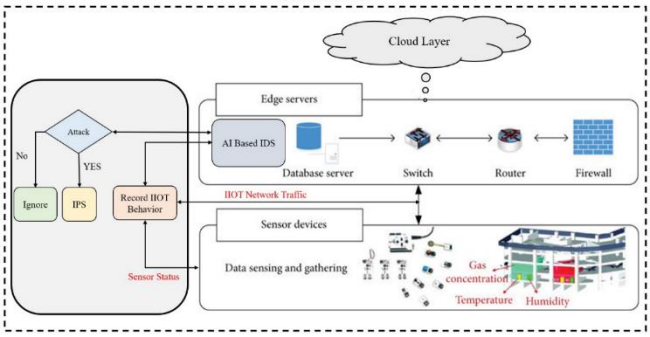


Fig 1: Components AI-Based IDS framework**.**

Our proposed work consists of several steps as shown in Figure 2, as follows:

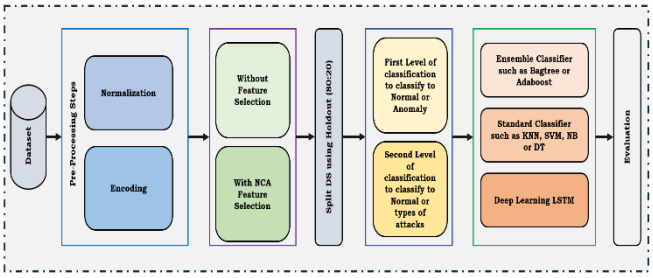


Fig 2: Proposed model of our smart IDS.

Step 1: Select dataset, and in our work, IoTID20 dataset [15] will use which consists of 86 columns including various network and flow-based features and 625,783 records. These columns capture different aspects of network behavior, which are essential for analyzing traffic and detecting anomalies in IoT environments. Knowing that three of these columns are for labels, which classify each record as normal or as one of several types of attacks as shown in Figure 3. The labels are critical for supervised learning tasks where the model needs to learn from labeled examples to predict the category of new, unseen data.

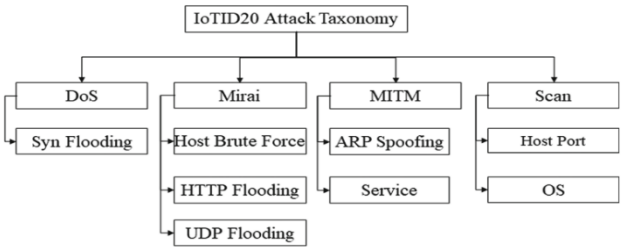


Fig 3: IoTID20 Attack Taxonomy

Step 2: Pre-Processing for preparing dataset to be suitable for training smart models and mainly consists of three steps:

- One-Hot Encoding: Machine learning models generally work better with numerical input. Categorical data are transformed into a numerical format that models can interpret without introducing ordinality.
- Min-Max Normalization: To scale numerical features in the dataset to a common scale without distorting differences in the ranges of values. This rescales the feature to a fixed range of 0 to 1, using the equation (1):

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (1)$$

Step 3: Feature Selection using NCA: To enhance the performance of nearest neighbors' classifiers by selecting and weighting features that contribute most to distinguishing between classes. NCA learns a linear transformation (a weight matrix) that maximizes the likelihood that a stochastic nearest neighbor will belong to the same class. This effectively learns which features are important by focusing on reducing distances between similar pairs (same class) and increasing distances between dissimilar pairs (different classes).

Step 4: Split DS using Holdout: The dataset is split into training (80%) and testing (20%) sets. The training set is used to build and train the model, while the testing set is used purely for evaluating its performance, simulating how the model would perform on unseen data.

Step 5: Classification: In our work we use two levels of classification:

- First Level - Normal or Attack: Determine whether each instance in the dataset is a normal operation or an anomalous attack.
- Second Level - Type of Attack: Further classify each detected attack into types, for instance, DOS, malware, or phishing, based on the characteristics of the attack.

Regarding classifier, we used three types of classifiers which will train using training datasets.

- Ensemble Classifiers: Use methods like Bagging Trees and AdaBoost, which combine the predictions of several base estimators to improve robustness and accuracy over a single estimator.
- Standard Classifiers: Implementing algorithms such as Support Vector Machine, K Nearest Neighbor, Naive Bayes, and Decision Trees, each having their particular strengths in the handling of different types of data and classification problems.
- Deep Neural Networks: Take advantage of the sophisticated models such as LSTM, which are very good at handling sequences and time series data,

thus, making them ideal for time dependent data anomaly detection.

Step 6: Evaluation: The most significant purpose of this measure is to determine the performance of the model on new, unseen data, which reflects its practical usefulness and accuracy. This is achieved through the application of different metrics including accuracy, precision, recall, F1-Mesaure, and ROC curves to assess different aspects of performance of the model to ensure it meets the required deployment standards.

The step-by-step instructions above are altogether the strong pipeline and outcomes framework for data handling, processing, and analysis for security objectives.

A. Neighborhood Component Analysis (NCA) For Feature selection

Neighborhood Component Analysis (NCA) is a dimensionality reduction and feature selection technique widely used to improve the performance of nearest neighbor classification. NCA works by learning a linear transformation of the dataset. NCA aims to transform the feature space so that the probability of a stochastic nearest neighbor belonging to the same class is increased. This is done by minimizing a loss function which sums over all misclassified examples weighted by their distance from the query point [16].

The approach includes training a weight matrix that rescales the feature space. What NCA tries to do by optimizing this matrix is to reduce the distances of data points that are similar (same class) and increase the distances between data points that are different (different classes). The procedure applies gradient descent, or other related optimization algorithms to achieve the optimal weights. This has enabled NCA to significantly improve the efficiency of nearest neighbor algorithms in the optimization of feature spaces, and for this, NCA has found use in a variety of machine learning applications, especially when it comes to both classification and clustering tasks [17].

B. Classification Algorithm

Adaboost (Adaptive Boosting) [18] is an ensemble technique, that uses several weak learners to generate a highly accurate classifier. Every learner, a simple decision tree usually, is added sequentially to the ensemble and focuses on the instances that were misclassified by the previous ones. The learners are assigned a weight by their accuracy, and also each instance in the dataset gets a weight that will be increased if it is misclassified. This makes the ensemble to be able to fit in the hard cases in the dataset. AdaBoost is very valuable for binary classification problems, and it is often noted for its ability to enhance classification accuracy.

Bagging Trees (BagTree) or Bootstrap Aggregating [19], is another ensemble method, that employs multiple decision trees trained on different sub-samples of the same training set. The subsets are generated by the process of random sampling of the training set with replacement. Each tree is trained individually and then their predictions are combined usually by majority voting for classification or averaging for regression. This approach decreases variance and eliminates overfitting, thus making an ensemble of decision trees more resistant as compared to individual trees.

SVM [20] is a strong classification method that operates by identifying the hyperplane that separates two classes of data with the maximum margin. Put simply, it identifies the largest "street" between classes. SVM is very efficient in high-dimensional spaces and is flexible enough to deal with linear as well as non-linear boundaries employing the trick of kernel.

K-Nearest Neighbors (KNN) is a simple, instance-based learning algorithm [21]. In KNN, the classification of a new instance is determined by a plurality vote of its neighbors, with the instance being assigned to the class most common among its k nearest neighbors measured by a distance function. KNN is easy to implement and understand but can become computationally expensive as the size of the data grows.

Decision Tree (DT) classifiers [22] are intuitive models that split data by learning decision rules inferred from the features. Trees are formed by nodes representing tests on features and leaf nodes representing classes. Decision trees are easy to interpret and can handle both numerical and categorical data but are prone to overfitting, especially with complex trees.

Naive Bayes (NB) classifiers [23] are probabilistic models that apply Bayes' Theorem, assuming strong (naive) independence between the features. They are particularly well-suited for classification tasks where high dimensionality is present, such as text classification. Despite their simplicity and the naive assumption, Naive Bayes classifiers often perform very well under many complex real-world situations.

Long Short-Term Memory (LSTM) [24] networks are a type of recurrent neural network (RNN) suitable for sequence prediction problems. Unlike standard feedforward neural networks, LSTMs have feedback connections and can process data sequences irrespective of input sequence length They are especially handy in tasks where context of the input data is of high importance, such as, speech recognition or anomaly detection in time series data. LSTMs prevent the vanishing gradient problem of the traditional RNNs, which makes them capable of learning longer dependencies.

Our LSTM model configuration is designed to effectively handle sequence classification tasks and consists of following layers:

- Input Layer: The model accepts sequences, where each sequence element corresponds to one feature.
- LSTM Layer: The LSTM layer has 200 hidden units, providing it with substantial capacity to learn from the data by capturing complex temporal dependencies within the sequence. It's configured to output only the last hidden state, which is typical for sequence-to-label tasks where the final state represents the culmination of learned temporal features relevant to the prediction task.
- Fully Connected Layer: This layer has as many neurons as there are unique classes in the dataset, ensuring each class can be predicted.
- Softmax Layer: Following the fully connected layer, the softmax layer normalizes the output to a probability distribution over the predicted classes, making the model's output interpretable as class probabilities.
- Classification Layer: The final layer computes the cross-entropy loss during training, which helps in optimizing the model by comparing the predicted output with the true class labels.

The LSTM model utilizes the Adam optimizer with settings optimized for GPU execution. Training is now set to 10 epochs at a learning rate of 0.001. Verbose output and progress plots are enabled to monitor the training process. This setup strikes a balance between achieving convergence in a reasonable number of training cycles and providing detailed insights into training progress through visual feedback.

Each of these models has its strengths and particular contexts where it excels as shown in Table 2, making them suitable for various kinds of data and predictive modelling challenges.

IV. PERFORMANCE MEASURES

Classification accuracy serves as an effective metric to gauge learning performance of proposed models [25-27]. The evaluation utilizes a standard confusion matrix technique, with the following metrics listed below:

Confusion Matrix: A confusion matrix provides a concise summary of prediction outcomes in classification problems. It offers insight into classification accuracy by distinguishing between correct and incorrect predictions as well as errors made within each class, distinguishing between true positives (true negatives) and misclassifications (false positives and false negatives). Diagonal elements indicate this information while off-diagonal ones represent misclassification errors like false positives/false negatives respectively.

Accuracy can be defined as the percentage of correctly classified samples when measured values are compared with known ones; expressed using equation (2):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2)$$

Where true positive (TP), true negative (TN), false positive (FP), and false negative (FN) represent key components of confusion matrices.

Table 2: Advantage and Disadvantage of each used model.

Classifier	Advantages	Disadvantages
AdaBoost	Enhances the accuracy of weak learners, robust to overfitting, less prone to overfitting than DT.	Sensitive to noisy data and outliers, performance depends on data and weak learner.
Bagging Trees	Reduces variance, less prone to overfitting, works well for high dimensional data.	Computationally intensive, less interpretable, not the best choice for very large datasets.
SVM	Effective in high-dimensional spaces, works well with clear margin separation, versatile (kernel methods).	Requires full data in memory, can be slow to train, choosing a correct kernel can be challenging.
KNN	Simple and effective, no training period, naturally handles multi-class cases.	Slow query time as dataset grows, sensitive to irrelevant features, needs homogeneous features.
Decision Trees	Easy to interpret and explain, can handle both numerical and categorical data.	Prone to overfitting, can be unstable, sensitive to small changes in the data.

Naive Bayes	Fast and efficient, performs well with large datasets, good baseline for text classification.	Assumes feature independence, poor estimates of probability can be a drawback.
LSTM	Excellent for sequential data, can model time series data, capable of learning long-term dependencies.	Computationally intensive, difficult to train, and can suffer from overfitting without proper regularization.

Error Rate measures the proportion of misclassified samples within the dataset as defined by equation (3). Ultimately, lower error rates mean better model performance.

$$Error\ Rate = \frac{FP + FN}{TP + FP + FN + TN} \quad (3)$$

Precision measures the model's ability to correctly classify positive values as shown by equation (4):

$$Precision = \frac{TP}{TP + FP} \quad (4)$$

Recall is the measure of the fraction of true positives (instances that are properly classified as belonging to a particular class) out of all positives, that is both true positives and false positives together. As shown by equation (5):

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \quad (5)$$

Specificity refers to a model's ability to predict negative values accurately as indicated by equation (6):

$$Specificity = \frac{TP}{TN + FP} \quad (6)$$

In the initial experiment, the details of the full dataset along with its training and testing splits is shown in Figure 4. The metrics used for comparison will include accuracy, error rate, precision, recall, specificity, and the F-measure across both the training and testing phases. This setup will allow us to assess the impact of NCA on model effectiveness comprehensively.

One of the evaluation metrics for binary classification models is **Area Under the Curve (AUC)**. It quantifies the area under a Receiver Operating Characteristic curve, which plots true positive rates versus false positive rates at different classification thresholds; an AUC of 1 represents perfect classification, whereas 0. 5 or less represents random classification.

Receiver Operating Characteristic (ROC) curves are graphical presentations of True Positive Rate versus False Positive Rate in binary classification models. Their location in the left upper corner is usually associated with good model performance.

F Measure is a frequent performance metric in binary classification assignments. It is the harmonic mean of precision and recall - an evaluation that takes both aspects into account. The calculation process of the F1 score is illustrated by the Equation (7):

$$\text{F Measure} = 2 \times \left(\frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}\right) \quad (7)$$

Training time is the duration required to train a model on a specific dataset. Fast training times are desirable as they facilitate rapid model development and deployment.

Testing time pertains to the duration taken to assess a trained model on a new dataset. Efficient testing times are advantageous as they enable prompt inference of model predictions.

V. RESULTS AND DISCUSSION

In this section, we conduct two experiments: the first experiment focuses on classifying network behavior as normal or anomalous, and the second experiment extends the classification to distinguish between normal operations and various types of attacks. For both experiments, we will evaluate different models with and without the application of NCA feature selection.

A. First Experiment:

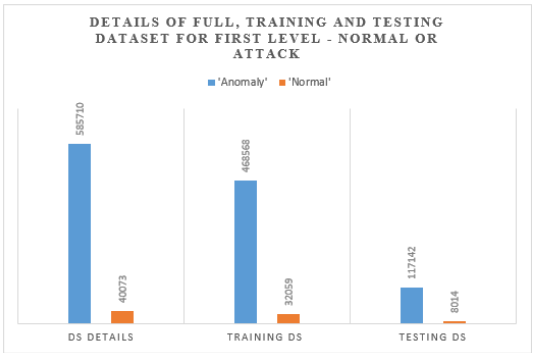


Figure 4: Details of Full, Training and testing datasets for first level of classification.

After applying NCA on the dataset, we can specify specific score threshold for dropping the columns which their score less than threshold. Figure 5 illustrates the feature scores for each column using NCA, both in their original order and ranked from highest to lowest importance. These visuals provide valuable insights into which features are most influential in the dataset when NCA is applied.

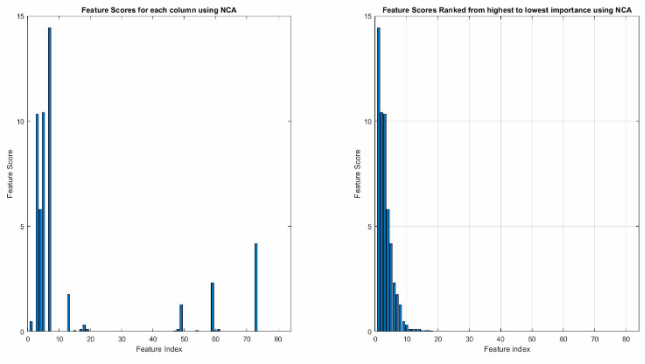
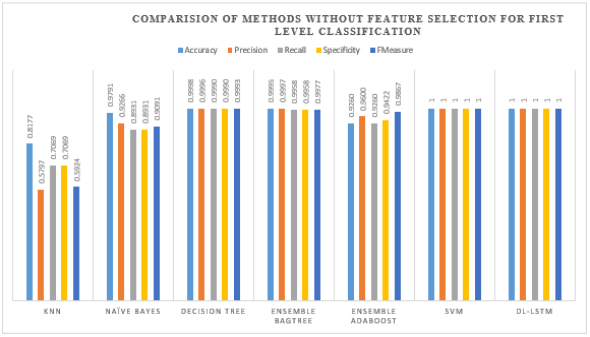


Fig 5: The feature scores for each column using NCA, both in their original order and ranked from highest to lowest importance, for first level of classification.

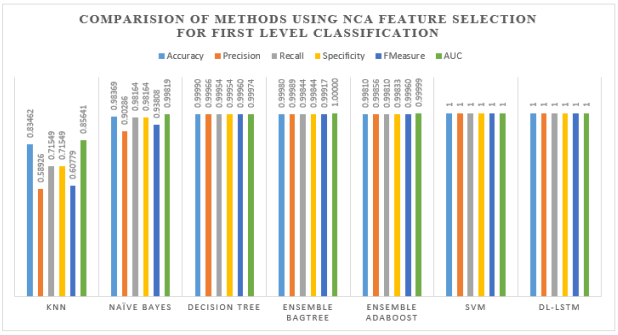
Knowing that, setting a threshold value for feature scores in NCA can significantly influence model performance and computational efficiency. Our choice of a threshold value of 0.0022, which results in selecting the top 20 highest-scoring features and dropping the other 63, appears to be a strategic compromise between performance and efficiency.

Knowing that, choosing a higher threshold value will lead to fewer features being selected. This can simplify the model further and reduce training and testing times, but at the risk of dropping potentially important features, which might degrade the model's performance. It's crucial to ensure that the selected features still capture the essence of the data without sacrificing the ability to generalize well. Whereas, a lower threshold value means more features will be retained, which might maintain or even slightly enhance performance due to the richer feature set. However, this comes with increased computational costs, as more features require more processing power and time during both training and testing phases. This might not be desirable, especially in scenarios where rapid decision-making is critical. The columns that their score high or equal threshold ranking from highest to lowest score are: Timestamp, Dst_Port, Src_Port, Dst_IP, Init_Bwd_Win_Byts, Pkt_Size_Avg, Fwd_Pkt_Len_Max, Pkt_Len_Var, Flow_ID, Bwd_Pkt_Len_Min, Bwd_Pkt_Len_Mean, Bwd_Seg_Size_Avg, Pkt_Len_Std, Bwd_Pkt_Len_Max, Fwd_Pkt_Len_Mean, Fwd_Seg_Size_Avg, ACK_Flag_Cnt, Pkt_Len_Mean, Pkt_Len_Max, Bwd_Pkt_Len_Std.

The results of the first experiment are shown in Figure 6 and Table 3 where (a) specific for comparing different model without using NCA, whereas (b) is for results with using NCA feature selection.



a



b

Fig 6: Results of the first experiment for binary classification (a) without and (b) with NCA feature selection.

From Figure 6 and table 3, we can observe several key insights based on the metrics provided as follows:

- Without NCA, models like SVM and DL-LSTM achieve perfect scores (1.00) across all metrics. Ensemble methods such as Decision Tree and Ensemble BagTree also display near-perfect performance, suggesting a strong fit.
- With NCA, the performance of most models either improves slightly or remains statistically consistent, particularly in precision, recall, and F Measure metrics.
-

Table 3: Results of first experiment.

3.a Results of the different models without NCA Feature Selection -First Level

Methods	Accuracy	Precision	Recall	Specificity	F Measure	AUC	Error Rate	Training Time (sec)	Testing Time (sec)
KNN	0.81769	0.57975	0.70685	0.70685	0.59241	0.85387	0.18231	8.02	0.45
Naïve Bayes	0.97906	0.92662	0.89314	0.89314	0.90909	0.98120	0.02094	8.99	3.21
Decision Tree	0.99983	0.99956	0.99904	0.99904	0.99930	0.99894	0.00017	34.32	0.25
Ensemble BagTree	0.99946	0.99971	0.99576	0.99576	0.99772	1.00000	0.00054	400.82	3.59
Ensemble Adaboost	0.92598	0.96002	0.92598	0.94224	0.98666	0.99999	0.98814	78.30	0.43
SVM	1	1	1	1	1	1	0	669.97	2.43
DL-LSTM	1	1	1	1	1	1	0	319.46	11.42

3.b Results of the different models with NCA Feature Selection -First Level

Methods	Accuracy	Precision	Recall	Specificity	F Measure	AUC	Error Rate	Training Time (sec)	Testing Time (sec)
KNN	0.83462	0.58926	0.71549	0.71549	0.60779	0.85641	0.16538	1.44	0.12
Naïve Bayes	0.98369	0.90286	0.98164	0.98164	0.93808	0.99819	0.01631	1.25	0.40
Decision Tree	0.99990	0.99966	0.99954	0.99954	0.99960	0.99974	0.00010	5.98	0.09
Ensemble BagTree	0.99980	0.99989	0.99844	0.99844	0.99917	1.00000	0.00020	152.51	2.40
Ensemble Adaboost	0.99810	0.99856	0.99810	0.99833	0.99960	0.99999	0.0019	13.61	0.13
SVM	1	1	1	1	1	1	0	182.36	1.63
DL-LSTM	**1**	**1**	**1**	**1**	**1**	**1**	**0**	160.56	6.42

- Notably, KNN shows significant improvements in accuracy and precision, which suggests that NCA helps focus this model on more relevant features, thereby enhancing its effectiveness.
- Further analysis reveals Naive Bayes exhibiting a slight improvement in most metrics with NCA, maintaining good performance even without it—this indicates robustness to feature selection.
- Both Decision Tree and Ensemble BagTree methods maintain high performance with and without NCA, suggesting their inherent ability to manage irrelevant or redundant features effectively.
- SVM and DL-LSTM continue to show strong performance in both scenarios. Additionally, there is a general decrease in error rates with the application of NCA, indicating fewer misclassifications, and the AUC remains high, showcasing excellent model capability in distinguishing between classes.
- Training times for models like KNN are notably reduced with NCA, from 8.02 seconds to 1.44 seconds, improving both model accuracy and operational efficiency. Testing times also see reductions, which benefits real-time prediction scenarios, making NCA a valuable preprocessing step for deployment in performance-sensitive environments.

Overall, the application of NCA feature selection appears to enhance model performance or maintain high performance while reducing computational overhead, emphasizing its importance in managing complex datasets like IoTID20.

B. Second experiment:

In the second experiment, which mirrors the first in methodology, the focus shifts to classifying data into categories of 'normal' or various 'types of attack.' The experiment utilizes NCA with a threshold value of 0.0016, strategically selecting the top 28 features for the classification task. Detailed distributions of the full dataset along with the training and testing splits are outlined in Figure 7.

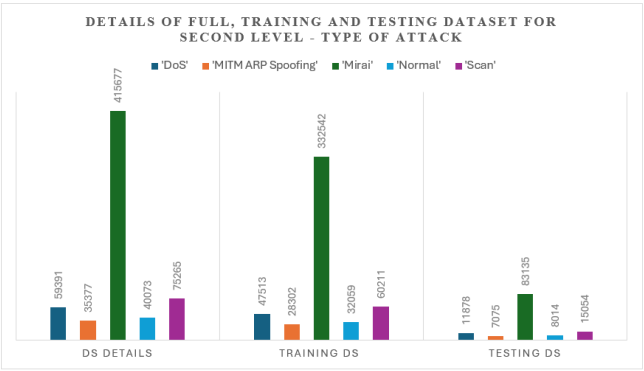


Fig 7: Details of Full, Training and testing datasets for second level (Multiclass) of classification**.**

Additionally, the impact of applying NCA with the specified threshold is illustrated in Figure 8, highlighting how this approach refines feature selection to enhance model performance in distinguishing between normal operations and potential attacks.

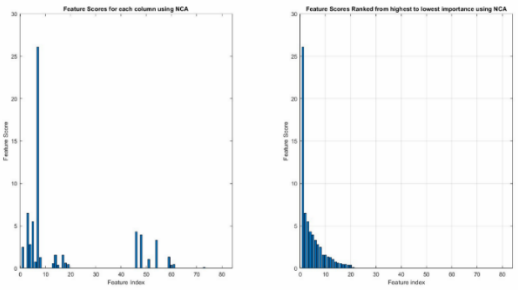
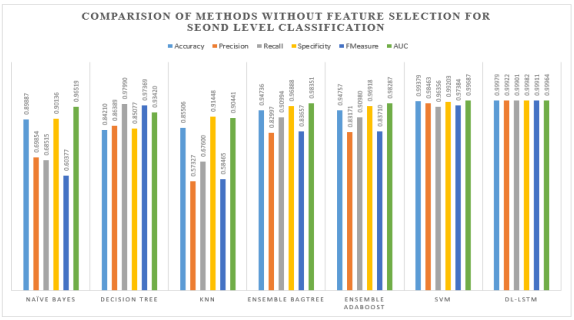


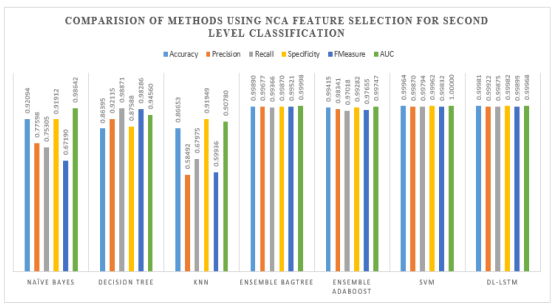
Fig 8: The feature scores for each column using NCA, both in their original order and ranked from highest to lowest importance for second level of classification.

The columns that their score high or equal threshold ranking from highest to lowest score are: Timestamp, Src_Port, Dst_Port, Pkt_Len_Max, Pkt_Len_Std, ACK_Flag_Cnt, Dst_IP, Flow_ID, Fwd_Pkt_Len_Min, Bwd_Pkt_Len_Max, Pkt_Size_Avg, Flow_Duration, SYN_Flag_Cnt, Protocol, Bwd_Pkt_Len_Min, Fwd_Pkt_Len_Max, Bwd_Pkt_Len_Mean, Bwd_Seg_Size_Avg, Fwd_Pkt_Len_Mean, Fwd_Seg_Size_Avg, Init_Bwd_Win_Byts, Bwd_IAT_Tot, Src_IP, Pkt_Len_Mean, Fwd_Pkt_Len_Std, Pkt_Len_Var, Pkt_Len_Min, Idle_Max.

The results of the second experiment are shown in Figure 9 where (a) specific for comparing different model without using NCA, whereas (b) is for results with using NCA feature selection. Moreover, the training and testing time for second level of classification is shown in table 4.



a



b

Fig 9: Results of second experiment for multiclass classification (a) without and (b) with NCA feature selection.

From Figure 9 we can observe the following:

- The use of NCA generally enhances model performance across most metrics. This is especially evident in models like SVM and DL-LSTM, where almost all performance metrics are at or near perfect scores.
- Notably, the error rates for most models are significantly reduced when NCA is applied. This suggests that NCA effectively enhances the models' ability to generalize, which is critical in avoiding overfitting and improving model robustness.
- Naïve Bayes: Shows improvement in accuracy, precision, and AUC with NCA, reflecting better generalization and effective feature utilization.
- Decision Tree: This model shows a high recall rate consistently, but a minor rise in error rate occurs with NCA, which implies trade-off between sensitivity and specificity.
- KNN: With the use of NCA only a moderate improvement in performance metrics is observed, showing that KNN does benefit from feature selection, but its performance is limited when compared to more sophisticated models.
- Ensemble BagTree and Adaboost: Both models demonstrate large gains in all measures with NCA, especially in AUC and error rates, that confirm the strength of ensemble methods with proper feature selection.
- SVM: NCA achieves almost perfect metrics in all dimensions, thus, being one of the best performing in both setups.
- DL-LSTM: Keeps a very good performance metrics with and without NCA, with small improvements in error rates and AUC with NCA, demonstrating its ability to handle complex data patterns.

This detailed study emphasizes the usefulness of NCA in improving classification models performance, which makes it a useful tool for the preprocessing stage of model training, particularly when it comes to complex and big datasets.

From Table 4, we can observe that:

- In absence of NCA, models such as Naïve Bayes and Decision Tree show short training and testing times, efficiency and fast evaluating, respectively. KNN gives moderate times but is a little bit slower than the other two. The more complicated models, like BagTree, Adaboost, SVM, and DL-LSTM, from the ensemble methods, have long training times because of their intricate computations and dealing with high-dimensional data. More specifically, DL-LSTM results in the longest training and testing times which point to its computational intensity

Table 4: Time for training and testing of different models for the second level of classification with and without NCA.

	Training Time (sec)		Testing Time (sec)	
Methods	Without NCA	With NCA	Without NCA	With NCA
Naïve Bayes	6.62	1.77	3.63	1.17
Decision Tree	28.85	10.02	0.18	0.08
KNN	7.35	2.66	0.77	0.29
Ensemble BagTree	454.98	222.48	5.83	4.27
Ensemble Adaboost	871.40	363.81	3.54	3.56
SVM	133.52	66.33	3.54	3.23
DL-LSTM	505.59	177.64	16.20	7.31

Table 5: Performance measure for each class for LSTM using NCA Feature Selection

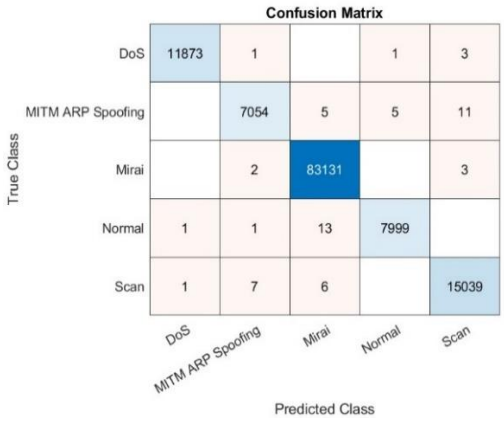
Class	Accuracy	Precision	Recall	Specificity	F Measure	AUC	Error Rate
'DoS'	0.99994	0.99983	0.99958	0.99998	0.99971	0.99987	0.00006
'MITM ARP Spoofing'	0.99974	0.99844	0.99703	0.99991	0.99774	0.99949	0.00026
'Mirai'	0.99977	0.99971	0.99994	0.99943	0.99983	0.99981	0.00023
'Normal'	0.99983	0.99925	0.99813	0.99995	0.99869	0.99943	0.00017
'Scan'	0.99975	0.99887	0.99907	0.99985	0.99897	0.99977	0.00025
Average	0.99981	0.99922	0.99875	0.99982	0.99899	0.99968	0.00019

because it is based on an architecture optimized for sequential data. The overall comparison emphasizes the computational requirements differences among various classification models.

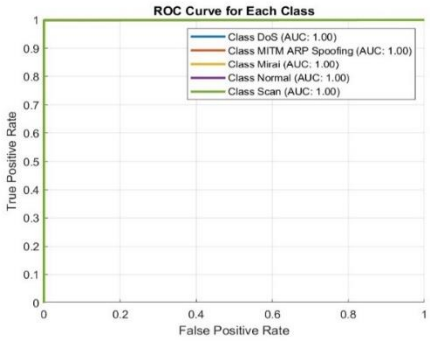
- The use of NCA always results in lower training and testing times for all models. This implies that, by reducing the number of features, NCA does not only benefit the computational process but may also help in concentrating the model's learning on the most influential features, therefore, improving the model accuracy and response.
- Models such as DL-LSTM enjoy the most important advantage from NCA in the form of time-saving that is crucial for deployment in environments when time is of the essence.
- Although ensemble methods still take a significant amount of computational time even with NCA, the decrease is substantial, which can make them more practical in scenarios where their robustness can be used without such a heavy penalty on performance speed.

In general, NCA helps in improving computational efficiency and is especially useful when optimizing complex models for real-time or near real-time applications in network behavior classification.

Finally, the performance measure for each class for LSTM using NCA Feature Selection, confusion matrix and ROC curve are presented in Table 5, Figure 10-a and Figure 10-b respectively.



a



b

Fig 10: (a) Confusion matrix and (b) ROC curve for LSTM using NCA for second level of classification.

VI. CONCLUSION

This study showcases the high potential of combining Machine Learning (ML) and Deep Learning (DL) methods with Neighborhood Component Analysis (NCA) for improving Intrusion Detection Systems (IDS) in Industrial Internet of Things (IIoT) environments. Extensive experimentations have supported the fact that ML and DL models, when optimized using NCA feature selection, not only show improvements in performance metrics such as accuracy, precision, recall, and AUC but also demonstrate significant reductions in training and testing times. Of special note is the efficiency of NCA in reducing the dimensionality of the features without losing the capability of detection. It facilitates better data processing that is vital in real-time intrusion detection scenarios, with prompt response being a requirement. The models tested, particularly the DL-LSTM and SVM, have yielded results close to perfection scores, which demonstrate the strength of the approach in a security-critical setting.

For future research in improving Intrusion Detection Systems for the Industrial Internet of Things (IIoT), the focus should be on the combination of advanced machine learning techniques among which are hybrid models that combine different ML and DL approaches to effectively deal with complex attack patterns. With federated learning, the privacy and scalability of such systems in distributed settings could be enhanced. Further, the improvement of autonomous and adaptive IDS capabilities that will be able to update and respond to new threats on the fly would be advantageous. Increasing the testing to wider and bigger datasets would enable the verification of the models' robustness and scalability. Finally, integration of XAI practices would improve the transparency and trustworthiness of such systems, a key aspect for their operational compliance in the context of sensitive industrial applications.

References

[1] Alotaibi, B. (2023). A survey on Industrial Internet of Things security: Requirements, attacks, AI based solutions, and edge computing opportunities. Sensors, 23(17), 7470. https://doi.org/10.3390/s23177470

[2] Mirani, A. A., Velasco-Hernandez, G., Awasthi, A., & Walsh, J. (2022). Key challenges and emerging technologies in Industrial IoT architectures: A review. Sensors, 22(15), 5836. https://doi.org/10.3390/s22041586

[3] Oladimeji, D., Gupta, K., Kose, N. A., Gundogan, K., Ge, L., & Liang, F. (2023). Smart transportation: An overview of technologies and applications. Sensors, 23(8), 3880. https://doi.org/10.3390/s23083880

[4] Gupta, K., Oladimeji, D., Kose, N. A., Gundogan, K., Ge, L., & Liang, F. (2023). The role of Industrial IoT in manufacturing for implementation of smart industry. Sensors, 23(21), 8958. https://doi.org/10.3390/s23218958

[5] Yao, H., Gao, P., Zhang, P., Wang, J., Jiang, C., & Lu, L. (2019). Hybrid Intrusion Detection System for Edge-Based IIoT Relying on Machine-Learning-Aided Detection. IEEE Network, 33, 75-81. https://doi.org/10.1109/MNET.001.1800479.

[6] Eid, A., Nassif, A., Soudan, B., & Injadat, M. (2023). IIoT Network Intrusion Detection Using Machine Learning. 2023 6th International Conference on Intelligent Robotics and Control Engineering (IRCE), 196-201. https://doi.org/10.1109/IRCE59430.2023.10255088.

[7] Zolanvari, M., Teixeira, M., Gupta, L., Khan, K., & Jain, R. (2019). Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things. IEEE Internet of Things Journal, 6, 6822-6834. https://doi.org/10.1109/JIOT.2019.2912022.

[8] Guezzaz, A., Azrour, M., Benkirane, S., Mohy-Eddine, M., Attou, H., & Douiba, M. (2022). A Lightweight Hybrid Intrusion Detection Framework using Machine Learning for Edge-Based IIoT Security. Int. Arab J. Inf. Technol., 19, 822-830. https://doi.org/10.34028/iajit/19/5/14.

[9] Ferrag, M., Friha, O., Hamouda, D., Maglaras, L., & Janicke, H. (2022). Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications for Centralized and Federated Learning. IEEE Access, PP, 1-1. https://doi.org/10.1109/ACCESS.2022.3165809.

[10] Ullah, S., Boulila, W., Koubâa, A., & Ahmad, J. (2023). MAGRU-IDS: A Multi-Head Attention-Based Gated Recurrent Unit for Intrusion Detection in IIoT Networks. IEEE Access, 11, 114590-114601. https://doi.org/10.1109/ACCESS.2023.3324657.

[11] Vaiyapuri, T., Sbai, Z., Alaskar, H., & Alaseem, N. (2021). Deep Learning Approaches for Intrusion Detection in IIoT Networks – Opportunities and Future Directions. International Journal of Advanced Computer Science and Applications, 12. https://doi.org/10.14569/IJACSA.2021.0120411.

[12] Tang, W., Li, D., Fan, W., Liu, T., Chen, M., & Dib, O. (2023). An Intrusion Detection System Empowered by Deep Learning Algorithms. 2023 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech), 1137-1142. https://doi.org/10.1109/DASC/PiCom/CBDCom/Cy59711.2023.10361315.

[13] Rodríguez, D., Okey, O., Maidin, S., Udo, E., & Kleinschmidt, J. (2023). Attentive transformer deep learning algorithm for intrusion detection on IoT

systems using automatic Xplainable feature selection. PLOS ONE, 18. https://doi.org/10.1371/journal.pone.0286652.

[14] Saxena, A., Pant, B., Alanya-Beltran, J., Akram, S., Bhaskar, B., & Bansal, R. (2022). A Detailed Review of Implementation of Deep Learning Approaches for Industrial Internet of Things with the Different Opportunities and Challenges. 2022 5th International Conference on Contemporary Computing and Informatics (IC3I), 1370-1375. https://doi.org/10.1109/IC3I56241.2022.10072499.

[15] I. Ullah and Q. H. Mahmoud, "A Scheme for Generating a Dataset for Anomalous Activity Detection in IoT Networks." In: Goutte C., Zhu X. (eds) Advances in Artificial Intelligence. Canadian AI 2020. Lecture Notes in Computer Science, vol 12109. Springer, Cham. https://doi.org/10.1007/978-3-030-47358-7_52

[16] Daniel, L., Chen, P.-Y., Liu, S., Ko, C.-Y., Mohapatra, J., & Weng, T.-W. (2022). Revisiting contrastive learning through the lens of neighborhood component analysis: An integrated framework. Proceedings of the 39th International Conference on Machine Learning. https://proceedings.mlr.press/v162/ko22a.html

[17] Ko, C.-Y., Mohapatra, J., Liu, S., Chen, P.-Y., Daniel, L., & Weng, T.-W. (2022). Revisiting contrastive learning through the lens of neighborhood component analysis: An integrated framework. MIT-IBM Watson AI Lab. Retrieved from https://mitibmwatsonailab.mit.edu/research/blog/revisiting-contrastive-learning-through-the-lens-of-neighborhood-component-analysis-an-integrated-framework/

[18] Tsiapoki, S., Bahrami, O., Häckell, M., Lynch, J., & Rolfes, R. (2020). Combination of damage feature decisions with adaptive boosting for improving the detection performance of a structural health monitoring framework: Validation on an operating wind turbine. *Structural Health Monitoring*, 20, 637 - 660. https://doi.org/10.1177/1475921720909379.

[19] Yu, H., Xu, C., Geng, G., & Jiang, Q. (2024). Multi-Time-Scale Shapelet-Based Feature Extraction for Non-Intrusive Load Monitoring. *IEEE Transactions on Smart Grid*, 15, 1116-1128. https://doi.org/10.1109/TSG.2023.3285117.

[20] Chen, L., Dong, X., Wang, B., Shang, L., & Liu, C. (2024). An Edge Computing-Oriented Islanding Detection Using Differential Entropy and Multi-Support Vector Machines. *IEEE Transactions on Smart Grid*, 15, 191-202. https://doi.org/10.1109/TSG.2023.3288361.

[21] Pujar, P., Kumar, A., & Kumar, V. (2024). Efficient plant leaf detection through machine learning approach based on corn leaf image classification. *IAES International Journal of Artificial Intelligence (IJ-AI)*. https://doi.org/10.11591/ijai.v13.i1.pp1139-1148.

[22] Wang, Y., Yan, Z., Sang, L., Hong, L., Hu, Q., Shahidehpour, M., & Xu, Q. (2024). Acceleration Framework and Solution Algorithm for Distribution System Restoration Based on End-to-End Optimization Strategy. *IEEE Transactions on Power Systems*, 39, 429-441. https://doi.org/10.1109/TPWRS.2023.3262189.

[23] Nordin, S., Wah, Y., Haur, N., Hashim, A., Rambeli, N., & Jalil, N. (2024). Predicting automobile insurance fraud using classical and machine learning models. *International Journal of Electrical and Computer Engineering (IJECE)*. https://doi.org/10.11591/ijece.v14i1.pp911-921.

[24] Vaiyapuri, T., & Binbusayyis, A. (2024). Deep self-taught learning framework for intrusion detection in cloud computing environment. *IAES International Journal of Artificial Intelligence (IJ-AI)*. https://doi.org/10.11591/ijai.v13.i1.pp747-755.

[25] Alrahhal, M., & Supreethi K.P. (2020). Multimedia Image Retrieval System by Combining CNN With Handcraft Features in Three Different Similarity Measures. International Journal Of Computer Vision And Image Processing, 10(1), 1-23. DOI: 10.4018/ijcvip.2020010101.

[26] Alrahhal, M., & K.P, S. (2021). Full Direction Local Neighbors Pattern (FDLNP). International Journal Of Advanced Computer Science And Applications, 12(1). DOI: 10.14569/ijacsa.2021.0120116.

[27] Alrahhal, M., & K P, S. (2021). COVID-19 Diagnostic System Using Medical Image Classification and Retrieval: A Novel Method for Image Analysis. The Computer Journal. DOI: 10.1093/comjnl/bxab051.