Advances in Science and Technology Research Journal

Advances in Science and Technology Research Journal 2023, 17(2), 120–135
https://doi.org/10.12913/22998624/162004
ISSN 2299–8624, License CC-BY 4.0

Received: 2023.02.12
Accepted: 2023.03.15
Published: 2023.04.01

Enhancing Intrusion Detection in Industrial Internet of Things through Automated Preprocessing

Anıl Sezgin[1,2*], Aytuğ Boyacı[3]

[1] ATASAREN, National Defence University, Konaklar, Yenilevent, Org. İzzettin Aksalur Cd., 34334 Beşiktaş, Istanbul, Turkey

[2] Siemens Corporate Technology, Esentepe, Yakacık Yolu No:111, 34870 Kartal, Istanbul, Turkey

[3] Department of Computer Engineering, National Defence University, Air Force Academy, Istanbul, Turkey

* Corresponding author's e-mail: anil.sezgin@siemens.com

ABSTRACT
Industrial Internet of Things (IIoT) is a rapidly growing field, where interconnected devices and systems are used to improve operational efficiency and productivity. However, the extensive connectivity and data exchange in the IIoT environment make it vulnerable to cyberattacks. Intrusion detection systems (IDS) are used to monitor IIoT networks and identify potential security breaches. Feature selection is an essential step in the IDS process, as it can reduce computational complexity and improve the accuracy of the system. In this research paper, we propose a hybrid feature selection approach for intrusion detection in the IIoT environment using Shapley values and a genetic algorithm-based automated preprocessing technique which has three automated steps including imputation, scaling and feature selection. Shapley values are used to evaluate the importance of features, while the genetic algorithm-based automated preprocessing technique optimizes feature selection. We evaluate the proposed approach on a publicly available dataset and compare its performance with existing state-of-the-art methods. The experimental results demonstrate that the proposed approach outperforms existing methods, achieving high accuracy, precision, recall, and F1-score. The proposed approach has the potential to enhance the performance of IDS in the IIoT environment and improve the overall security of critical industrial systems.

Keywords: feature selection, intrusion detection, automated machine learning, industrial internet of things

INTRODUCTION

The Industrial Internet of Things (IIoT) has revolutionized the way we think about industrial systems, bringing together physical devices, sensors, and software to enable real-time monitoring and control. By providing real-time data on production processes, supply chain management, and equipment maintenance, IIoT enables companies to optimize their operations, minimize downtime, and reduce costs. IIoT also allows for greater flexibility and agility in the face of changing market demands.

With access to real-time data, manufacturers can quickly adjust production processes and adapt to changing consumer preferences, without sacrificing quality or efficiency. This enables companies to respond more quickly to market trends and gain a competitive advantage. The increasing connectivity and complexity of these systems also create new challenges, such as the risk of cyber-attacks and intrusions. Intrusion detection systems (IDS) play a crucial role in protecting IIoT systems by detecting and mitigating potential security threats. An IDS is a security solution that monitors network traffic for signs of malicious activity. It works by analyzing network traffic and identifying patterns that are indicative of an attack. An IDS can help mitigate the cyber attack risks associated with IIoT by detecting and responding to threats in real-time [1].

IDS can be categorized into different types, such as anomaly-based, signature-based, and hybrid methods. Furthermore, anomaly-based methods can be subcategorized as statistical-based, knowledge-based, or machine learning-based. The statistical-based method uses a stochastic model of normal operation to compare traffic statistics and detect anomalies. Knowledge-based methods use expert rules to describe normal and attack behavior, while fuzzy-based methods use a rule-based system connected to input data. Machine learning-based techniques create an explicit or implicit model of observed patterns and update them regularly to improve intrusion detection efficiency.

Machine learning algorithms leverage the power of computer science and mathematics to develop models that can learn and improve upon themselves with experience. The process involves exposing the algorithms to vast amounts of data, allowing them to identify and learn from patterns that are too complex for traditional non-machine learning approaches. By using this approach, machine learning algorithms can make predictions, detect anomalies, and provide insights with a level of accuracy and precision that non-machine learning techniques cannot match. The ability to continuously learn and improve through exposure to new data is one of the key strengths of machine learning, which has made it an indispensable tool for a wide range of applications across numerous industries [2].

Numerous techniques exist to analyze network traffic flow, including stateful protocol analysis, anomaly detection, and misuse detection. Misuse detection involves using predefined signatures and filters to recognize attacks, which necessitates consistent human intervention to keep the signature database up to date. Although this technique can successfully identify known attacks, it may not be useful in detecting new or unknown attacks. In contrast, anomaly detection utilizes heuristic mechanisms to pinpoint malicious activities that deviate from normal behavior. However, it can lead to a high rate of false positives, particularly in complex environments.

Preprocessing is an essential step in IDS that involves transforming raw data into a format that is suitable for model training [3]. Preprocessing techniques such as imputation, scaling and feature selection can help to identify patterns and anomalies in the data, enabling the IDS to detect potential security threats more accurately and efficiently. However, the preprocessing stage can be time-consuming and resource-intensive, and can require domain-specific knowledge, which can limit the scalability and usability of the IDS.

To overcome these challenges, researchers have been exploring the use of automated preprocessing techniques that can streamline the data preparation process and improve the quality of the data used for model training. Automated preprocessing techniques leverage algorithms and statistical models to handle tasks such as missing data imputation, feature scaling, and outlier detection, without requiring extensive domain knowledge or manual intervention.

Related work

Cyberattacks on IIoT systems can have serious consequences, such as production downtime, equipment damage, loss of intellectual property, and even risk to human safety. To mitigate these risks, Intrusion Detection Systems (IDS) have become a crucial part of IIoT security infrastructure. IDS are designed to detect and respond to cyberattacks, and their implementation can greatly reduce the impact of cyberattacks on IIoT systems. However, developing an effective IDS for IIoT is a complex task due to the unique challenges posed by the complex and dynamic nature of IIoT systems.

In study [4], it is suggested that a deep neural network (DNN) could be utilized to design an intrusion detection system (IDS) that is both efficient and expandable. This IDS has the capability to identify and categorize cyberattacks automatically and promptly, at both network and host levels, using machine learning methods. The study highlights the challenges in using machine learning for IDS due to the continuously changing nature of cyberattacks and the need for systematic benchmarking of publicly available malware datasets. The study provides a comprehensive evaluation of various machine learning algorithms on multiple publicly available benchmark malware datasets, and demonstrates the superior performance of DNNs over classical machine learning classifiers. According to the study, the DNN model put forward can acquire abstract and high-dimensional feature representations of IDS data. The authors suggest a hybrid DNN framework that is scalable and can monitor network traffic and host-level events in real-time, enabling the system to alert potential cyber attacks proactively.

The purpose of the study [5] is to create a reliable and efficient technique for detecting fraud in internet-based transactions. With the rise in online transactions, the likelihood of fraudulent behavior has also increased. Detecting and preventing fraudulent transactions is critical for preserving the trust of customers in online payment systems. Traditional rule-based methods for detecting fraud are no longer effective due to the complexity and diversity of fraudulent activities. Therefore, researchers are turning to machine learning techniques to build fraud detection models that can learn from large amounts of data and detect fraudulent patterns. This research proposes a deep learning-based method for detecting fraud in internet-based transactions. The method is specifically designed to handle unbalanced and dynamic data and utilizes a combination of deep learning architectures such as convolutional neural networks and long short-term memory networks. The effectiveness of this method is demonstrated by evaluating its performance on a publicly available credit card fraud dataset. The results indicate that this approach outperforms existing machine learning techniques for fraud detection in terms of precision, recall, accuracy, and F1 score. This proposed approach has the potential to be utilized in real-world online payment systems to enhance their security and prevent fraudulent activities.

An attack detection model for computer networks is proposed in [6]. The study suggests a technique for selecting features based on correlation, which utilizes the Pearson correlation algorithm to identify features with a high degree of correlation. The selected features are used for classification modeling using Random Forest, and the results show high detection accuracy, precision, and recall. The proposed model can be used to improve existing IDS models.

In 2015, the UNSW-NB15 dataset was developed to aid in the research of intrusion detection, and a study [7] has utilized this dataset. The main objective of this study is to utilize machine learning techniques to identify significant features that can enhance intrusion detection in network systems, while also addressing the curse of high dimensionality caused by irrelevant and redundant features. To evaluate the efficacy of this feature selection approach, the proposed feature subset is compared with previous research on feature selection in the KddCup99 dataset. The lack of available research data for network intrusion detection is a major obstacle for researchers in this field, and the UNSW-NB15 dataset is a recent attempt to overcome this problem.

A new system for detecting network intrusions is introduced in [8]. The system employs a dual-action approach that includes meta-classifier-based feature selection and ensemble learning meta-classifiers for stacking via cross-validation to prevent overfitting. The goal of this system is to tackle the challenges faced by Intrusion Prevention Systems (IPS), which are a type of extended IDS network security technology that is designed to block threats and detect vulnerabilities in real-time while maintaining network performance. The study also points out the problems of computation cost and feature selection, which can reduce the effectiveness of IPS. Instead of utilizing features that are common to all types of attacks, the proposed system enhances the overall classification accuracy by selecting a more effective subset of informative features for each specific attack type. This accelerates the learning and testing process, leading to higher detection rates and fewer false alarms. Additionally, the system's ability to operate in online mode and scale is demonstrated through several experiments.

Researchers employ Genetic Algorithms to automate the process of rule generation for intrusion detection and prevention in Software Defined Networking (SDN) systems in [9]. Although SDN offers a flexible and customizable networking service by centrally managing the network, network attacks remain a significant concern for its development. Intrusion detection systems can detect and identify various threats, but predicting the formation of new attacks becomes challenging, making manual rule writing a tedious task. The study seeks to automate the rule generation process using Genetic Algorithm to predict the likelihood of new attack formation and generate rules for detection, prevention, and mitigation of their effects. The research specifically focuses on integrating intrusion detection systems with Genetic Algorithm to automate the rule generation process for predictive cases, addressing the difficulties posed by new and unknown network attacks in SDN systems.

A transfer learning intrusion detection system based on the Convolutional Neural Network architecture for cloud Internet of Things systems is presented in [10]. The ELETL-IDS model, which is designed for efficient and lightweight ensemble transfer learning, utilizes five pre-trained CNN models and the top three best-performing models to attain 100% accuracy in identifying network attacks. This model is both reliable and efficient, making it suitable for deployment in cloud IoT systems for intrusion detection purposes.

Purpose of study [11], is to focus on addressing the issue of missing values in applications, which can impact the accuracy of the results obtained through learning methods. The authors review existing solutions based on statistical or machine learning techniques, and discuss the limitations of these approaches. They propose the use of autoencoder networks and adversarial training as potential solutions, but note that current methods are impaired by network structure and training strategies. The authors argue that by carefully selecting network structure and optimization strategies, they can outperform other deep learning solutions, and demonstrate the impact of stochastic corruption of inputs on network performance.

Machine learning-based intrusion detection systems have been proposed to mitigate security risks, but they are often negatively affected by the imbalanced data distribution commonly found in IIoT environments [12]. The proposed solution in this paper addresses the issue of identifying unknown attacks by presenting a novel hybrid model called EvolCostDeep. This model combines stacked autoencoders and convolutional neural networks and uses a new cost-dependent loss function to optimize the model's parameters. The cost is determined using an evolutionary algorithm, which improves the accuracy of identifying unknown attacks. In addition to the EvolCostDeep model, the authors also introduce a fog computing-enabled framework called DeepIDSFog. This framework is designed to handle big Industrial Internet of Things traffic data and effectively addresses the class imbalance problem by parallelizing the EvolCostDeep model through task-level and model-level mechanisms. Overall, the proposed approach offers an effective solution for identifying unknown attacks in IIoT traffic data while also ensuring scalability and efficiency. The authors conducted a series of experiments on two different data sets, ToN-IoT and UNSW-NB15, and the results demonstrate that the proposed frameworks outperform other methods in terms of accuracy and speed. This paper contributes to the development of effective and scalable intrusion detection systems for IIoT environments, providing a promising solution to mitigate cyber attacks and intrusions in critical industries.

The authors propose using Artificial Intelligence techniques, particularly Deep Learning models, to enhance the performance of an IDS and reduce cyberattacks on IoT and IIoT networks. To achieve this, the authors employ an ensemble classifier method that combines multiple prediction algorithms to create a new model with greater precision in [13]. The authors incorporate an Artificial Neural Network, which updates all model weights based on training data to achieve optimal performance. Overall, the proposed approach aims to enhance the accuracy and efficiency of IDS through the use of advanced AI techniques in IoT and IIoT network settings. The proposed approach is evaluated on a dataset, and the results show that the stacked ensemble classifier outperforms any individual classifier, achieving an average accuracy of 99.7%. The authors make their code publicly available to ensure reproducibility. This paper represents a valuable contribution to the field of cybersecurity, offering a new approach to the challenging problem of intrusion detection and prevention.

An intrusion detection model that uses a combination of dimensionality reduction, chi-square feature selection and multi-class support vector machine (SVM) is developed in study [14]. The authors argue that using a sole classifier algorithm for intrusion detection is not sufficient, as it fails to achieve high attack detection rates with reduced false alarm rates, given the large amount of data to process. To address this problem, the proposed model reduces the data to an optimal set of attributes without losing information using dimensionality reduction techniques, then selects relevant features using chi-square feature selection. The reduced feature set is then used to train a multi-class SVM classifier, which has not been widely used for intrusion detection so far. The authors also apply a parameter tuning technique to optimize the SVM classifier's parameters, namely the gamma and overfitting constant. They experimentally evaluate the proposed approach on the NSL-KDD dataset, an enhanced version of the KDDCup 1999 dataset, and show that it results in better detection rates and reduced false alarm rates compared to existing approaches.

The growing necessity for intelligent decision-making in detecting cyber-attacks within the industrial internet of things (IIoT) context is discussed in [15]. While IIoT enables optimal industrial operations through the collection and processing of large amounts of data, it also creates new opportunities for malicious cyber-attacks. The paper proposes an intrusion detection system (IDS) that utilizes deep learning models to address this issue, specifically through the combination of long short-term memory (LSTM) and convolutional neural network (CNN) techniques, which have been proven to be effective in intrusion detection and classification. The proposed model is evaluated using the Edge-IIoTset dataset, which includes actual traffic networks of IoT and IIoT applications. To measure effectiveness, the model is compared to traditional machine learning models as well as recently proposed related models. The results of the study show that the CNN-LSTM model outperforms other models in terms of accuracy, demonstrating its effectiveness in detecting cyber security intrusions in IIoT applications.

A lightweight intrusion detection model is proposed in [16] for software-defined networks. The paper examines the vulnerabilities posed by the extensive connectivity and data exchange in the SDN-based industrial cyber-physical environment, with a particular emphasis on the SDN controller as a high-value target for cyberattacks. The paper highlights the difficulty of deploying intrusion detection systems based on deep learning in an IIoT network with stringent latency demands for prompt identification and mitigation of cyber attacks. To tackle this challenge, the paper introduces a lightweight intrusion detection model tailored to an SDN-based industrial CPS environment. The proposed model is assessed using a publicly available cyber-security dataset related to SDN, and it delivers high accuracy, precision, recall, and low time cost, surpassing existing state-of-the-art models.

An intrusion detection system (IDS) is presented as a monitoring and defense system against cyberattacks [17]. IDS uses signature-based and anomaly-based techniques to detect cyber attacks. However, these techniques face several challenges. Examples of these challenges are false alarms, complex and high-dimensional data, and long computation times. To address these challenges, the paper proposes using feature selection, specifically the correlation-based feature selection (CFS) approach, which reduces the amount of data processed by the IDS detection engine. The proposed CFS-based IDS is evaluated using the CIC-IDS2018 dataset and achieves high accuracy, recall, specificity, precision, F1-score, true positive rate, and true negative rate. The experimental results demonstrate that the proposed CFS-based IDS performs optimally, achieving an accuracy of 99.9995%.

In an effort to enhance the performance of intrusion detection systems, a novel network anomaly detection model, named SPIDER, has been proposed in a recent study [18]. This approach incorporates four types of Recurrent Neural Networks, namely Bi-LSTM, LSTM, Bi-GRU, and GRU, and utilizes Principal Component Analysis to reduce data dimensions, thus mitigating issues associated with high dimensionality. The primary goal of this study is to improve the accuracy and efficiency of IDSs for detecting network intrusions. The proposed model is assessed using two widely known datasets, NSL-KDD and UNSW-NB15, to evaluate its robustness and effectiveness in detecting intrusions. The findings reveal that the SPIDER model surpasses existing models in detecting intrusions and has the potential to make significant contributions to the development of more efficient intrusion detection systems.

A detection system for attacks in Industrial Internet of Things (IIoT) settings is introduced [19] using a Deep Random Neural Network (DRaNN). IIoT merges internet-connected smart devices and sensors, generating large amounts of data requiring intelligent processing for cybersecurity measures. The authors employed deep learning techniques to create their intrusion detection system, with the proposed DRaNN being a variation of the Artificial Neural Network that boasts improved generalization and a highly decentralized structure. To improve the accuracy of attack detection, the authors employed a combination of hybrid particle swarm optimization and sequential quadratic programming (SQP) to train the proposed RaNN, yielding optimal hyperparameters. The efficacy of the DRaNN-based scheme was evaluated by conducting experiments on three new IIoT datasets in both binary and multiclass configurations. The results showed superior performance compared to other attack detection methods in IIoT settings. This approach could contribute to the development of fast and reliable attack detection mechanisms for IIoT environments.

A novel technique has been suggested to improve the precision and detection rate of network-based intrusion detection systems (NIDS) for identifying anomalies while reducing the incidence of false positives [20]. The proposed approach utilizes a combination of artificial bee colony (ABC) algorithm for feature selection and the AdaBoost algorithm for the assessment and classification of features. Two datasets, namely NSL-KDD and ISCXIDS2012, were used to evaluate the efficiency of the proposed method, and the results indicate that it outperforms other IDS techniques in diverse attack scenarios in terms of accuracy and detection rate. The study concludes that the suggested hybrid method offers an effective solution to improve the performance of A-NIDS and reduce false alarms and detection accuracy issues that arise due to the large volume of network data.

The study [21] aims to evaluate the performance of different machine learning algorithms in detecting network intrusions using the NSL-KDD benchmark data set. It specifically focuses on using Support Vector Machine, J48, Random Forest, and Naïve Bytes algorithms for both binary and multi-class classification. The paper discusses the results of applying these algorithms and how they outperformed previous works in network intrusion detection. Overall, the paper aims to provide promising alternatives to traditional intrusion detection systems using machine learning and deep learning techniques.

METHOD

Automated preprocessing plays a vital role in detecting intrusions in IIoT systems. It allows the system to filter out irrelevant data, decrease data volume, and convert the data into an easy-to-analyze format. This leads to real-time detection of intrusions, enhances system efficiency, and guarantees the safety and security of IIoT systems.

Automated preprocessing holds importance for intrusion detection in industrial Internet of Things systems owing to various underlying reasons:

- Scale: IIoT systems generate massive amounts of data from various sources, including sensors, control systems, and other devices. This data needs to be preprocessed to filter out irrelevant information and reduce the data volume to a manageable size. Automated preprocessing techniques such as data reduction, sampling, and filtering can help in reducing the amount of data to be processed by an intrusion detection system, thus improving its efficiency and effectiveness.
- Timeliness: Timeliness is crucial in IIoT systems because any delay in processing data can result in serious consequences such as equipment failure, production loss, or even safety incidents. Automated preprocessing helps in real-time analysis of data, enabling quick detection and response to any intrusion attempts. By automating the preprocessing, the system can quickly filter and extract the essential features from the incoming data and identify any suspicious behavior in real-time.
- Complexity: IIoT systems are complex, and the data generated by them is often unstructured, noisy, and difficult to analyze. Automated preprocessing techniques such as data cleaning, feature extraction, and normalization help in transforming the data into a format that is easy to analyze. For instance, data cleaning can help remove missing or corrupted data, while feature extraction can help identify patterns and trends that can be used to detect anomalies in the data.
- Efficiency: Manual preprocessing of IIoT data can be time-consuming and error-prone. Automated preprocessing techniques help in reducing the time and effort required to preprocess data, thus increasing the efficiency of the intrusion detection system. By automating the preprocessing, the system can quickly filter out the irrelevant data and extract only the essential features needed to detect intrusions. This reduces the computational cost and speeds up the intrusion detection process.

Imputation

Missing data is a common issue in real-world datasets, and imputation is the process of replacing these missing values with estimated values based on other available information [22].

One of the primary advantages of imputation is that it can lead to more complete data. Without imputation, missing data would be excluded from the analysis, which could result in the loss of valuable information. Imputation can help make more use of the available data and prevent the exclusion of potentially relevant observations. Imputation can also lead to better statistical power. By imputing missing values, the resulting dataset will have a larger sample size. This can improve the statistical power of the analysis, making it easier to detect significant effects. Another advantage of imputation is that it can help reduce bias that may result from ignoring missing data. By estimating the missing values, imputation can help reduce the bias that may result from only analyzing complete cases. This can lead to more accurate results and better inference. Imputation can also help improve the accuracy of machine learning models. By imputing missing values, the resulting dataset is more complete and can be used to train more accurate models. This can lead to better predictions and more useful insights.

Imputation is a flexible technique that can be applied to a wide range of data types and can be used in various contexts and applications. Imputation methods can vary depending on the type of data and the context of the problem, making it a versatile tool for handling missing data. It is generally easy to use, with a wide range of available methods and software packages. This makes it a practical solution for handling missing data in many different research applications.

Some common types of imputation include:

- Mean imputation: Replacing missing values with mean of the non-missing values in the same column.
- Median imputation: Replacing missing values with the median of the non-missing values in the same column.
- Mode imputation: Replacing missing values with the mode of the non-missing values in the same column.
- Most frequent imputation: Replacing missing values with the most frequent values in the same column.

Scaling

In many machine learning algorithms, features with different scales can have a disproportionate impact on the model. Scaling refers to the process of transforming features in a dataset to have similar scales. This can help the algorithm converge faster and improve its accuracy.

Scaling can have a significant impact on the performance and accuracy of machine learning models. This is because some machine learning algorithms are sensitive to the scale of the input features. For example, algorithms like k-nearest neighbors, support vector machines, and neural networks use distance measures to calculate the similarity between data points. If the features have different scales, then the distance measure will be dominated by the features with the larger scale. This can lead to inaccurate predictions and suboptimal performance of the model.

Another impact of scaling on machine learning models is related to the convergence speed of iterative algorithms. Iterative algorithms such as gradient descent converge faster on scaled data, which leads to quicker convergence to the optimal solution. The choice of scaler depends on the nature of the dataset and the machine learning algorithm used. There are different types of scalers that are commonly used in machine learning, including:

- Min-Max Scaler: Min-max scaling is a method that transforms the features to a scale between 0 and 1. This method is useful when the features have a large range of values and some features may dominate others. Min-max scaling ensures that all features are on the same scale, making it easier for the machine learning algorithm to learn from the data. The method is simple to implement and has the advantage of preserving the original structure of the data. However, it can be sensitive to outliers, which can cause the feature values to be distorted. Min-Max scaling function is show in Eq. (1).

$$z_i = \frac{x_i - \min(x)}{\max(x) - \min(x)} \tag{1}$$

- Standard Scaler: Standard scaling is a method that transforms the features to have a mean of 0 and a standard deviation of 1. This method ensures that the features have the same mean and variance, making it easier for the machine learning algorithm to learn from the data. Standard scaling has the advantage of being easy to interpret and can be less sensitive to outliers than min-max scaling. However, it can be sensitive to outliers if the outliers are very large or very small. Standard scaling function is shown in Eq. (2).

$$z = \frac{x_i - \mu}{\sigma} \tag{2}$$

- Robust Scaler: Robust scaling is a method that transforms the features based on the median and interquartile range, making it robust to outliers. This method is useful when the data contains outliers, as it ensures that the outliers do not dominate the feature values. Robust scaling has the advantage of being robust to outliers and can provide better results when the data contains a large number of outliers. However, it can be more complex to implement and may not preserve the original structure of the data as well as min-max scaling or standard scaling. Robust scaling function can be seen in Eq. (3).

$$X_{new} = \frac{X - X_{median}}{IQR} \tag{3}$$

Feature selection

There are instances where a dataset may include numerous features that are not necessary for solving the problem at hand. To address this, feature selection involves the careful selection of a relevant subset of features from a larger pool within a dataset. By doing so, this approach can enhance model performance by mitigating the chances of overfitting and improving the model's comprehensibility.

Feature selection is critical in machine learning because it reduces the number of input features, which helps in reducing the computational complexity of the model. The inclusion of irrelevant features can lead to overfitting, where the model learns noise in the data rather than the underlying patterns. This results in poor performance on new and unseen data. Feature selection also helps in improving the interpretability of the model by reducing the number of input features.

There are three main types of feature selection techniques in machine learning:

- Filter methods: Filter methods select features based on their statistical properties such as correlation, variance, and mutual information. These methods are computationally efficient and can be used as a preprocessing step before applying a more complex machine learning algorithm.
- Wrapper methods: Wrapper methods evaluate the performance of the machine learning algorithm on different subsets of features. These methods are computationally expensive and require training and evaluating the model on multiple subsets of features.
- Embedded methods: Embedded methods incorporate feature selection as part of the machine learning algorithm. These methods learn the relevance of the features during the training phase of the algorithm.

The choice of feature selection technique depends on the nature of the dataset and the machine learning algorithm used. Filter methods are best suited for datasets with a large number of features and can be used as a preprocessing step before applying more complex algorithms. Wrapper methods are suitable for datasets with a small number of features and can be used when the computational cost is not a constraint. Embedded methods are suitable for algorithms like decision trees, random forests, and gradient boosting, which incorporate feature selection as part of the algorithm.

Experimental evaluation

Machine learning algorithms require a well-prepared and structured dataset to produce optimal results. Preprocessing the data prior to feeding it into a model is crucial to improve the accuracy and performance of the model. In this paper, we present a preprocessing pipeline that includes three steps: imputation, scaling, and feature selection. The pipeline is designed to handle missing data and make the features more suitable for machine learning algorithms. In our preprocessing pipeline, each step's output serves as an input for the next step, making the pipeline automated. This automation eliminates the need for manual intervention, ensuring consistency and reducing the risk of human error. The automated pipeline saves time and resources and makes the process of preparing data for machine learning more efficient. To evaluate effectiveness of each step and find the best result in pipeline, our model uses XGBoost classifier. The steps of automated preprocessing pipeline is shown in Figure 1.

Handling missing data is a common challenge in machine learning, as most algorithms do not work well with missing values. To address this issue, the first step in our pipeline is imputation, which replaces missing values with estimated values. In this step, we implement three imputation methods: mean imputation, median imputation, and most frequent imputation. Each imputation method is applied individually and the results are sent to the XGBoost classifier. In this way, it can be determined which imputation method gives the best result on the dataset.

The next step in our pipeline is scaling. The method that gives the best result in the imputation step is sent as input to the scaling step. Scaling is a crucial step in the preprocessing pipeline as many machine learning algorithms are sensitive to the scale of the features. This step transforms the features to a common scale so that they can be used in the machine learning algorithm. In our pipeline, we implement three scaling methods: min-max scaling, standard scaling, and robust scaling.

After each scaling method is applied, the XGBoost classifier is used to evaluate its effectiveness. The performance of the classifier is used as a measure of the effectiveness of each scaling method. The scaling method that provides the best performance is selected for use in the final model.

The final step in our pipeline is feature selection, which is the process of selecting a subset of

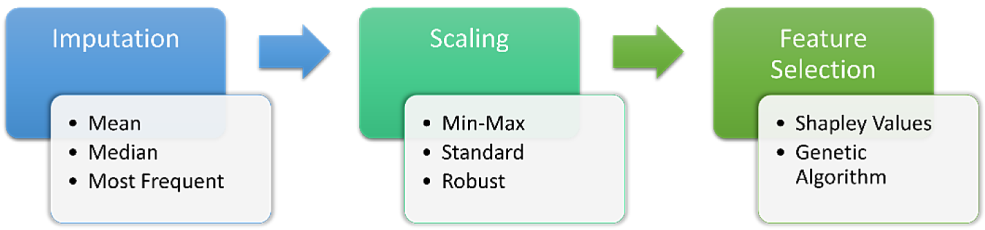

Figure 1. Steps of automated preprocessing pipeline

the features to use in a machine learning model. Feature selection is important because it reduces the complexity of the model, reduces overfitting, and improves the interpretability of the model. In this step, we implement a hybrid approach that combines shapley values and a genetic algorithm for feature selection.

Shapley values are a measure of feature importance that takes into account the contribution of each feature to the prediction made by the model. Our model uses shapley values to determine the importance of each feature and set a threshold for the feature importance. Two distinct methods for determining the threshold were employed in the study. The first approach involves setting a fixed value for the threshold, while the second approach involves computing the mean absolute Shapley value and using it as the threshold. This enables the acquisition of a generic threshold for each classification method and dataset. The mean absolute Shapley value is a measure of feature importance in Shapley. Shapley values represent the contribution of each feature to the prediction made by the model, and can be both positive and negative. The mean absolute Shapley value is calculated by taking the absolute value of the Shapley values for each feature, and then computing the mean value across all instances in the dataset. It provides a measure of the overall importance of each feature, taking into account both the magnitude and direction of the Shapley values. If a feature's importance is below the threshold, we do not use it in the genetic algorithm. The genetic algorithm is a optimization technique that uses principles of natural selection to find the optimal solution. In our case, the optimal solution is the subset of features that produces the best performance in the machine learning model. The genetic algorithm evaluates different combinations of features and selects the combination that results in the best performance. The optimal solution for selecting individuals with respect to feature selection is determined by the fitness function, which is used to evaluate the performance of each individual in the population. Fitness function is defined as the classification accuracy of XGBoost classifier on a given dataset using the selected subset of features. The genetic algorithm evolves the feature subset by repeatedly generating and evaluating candidate solutions until stopping criterion is reached.

To begin the genetic algorithm, a population is generated randomly. This population is then assessed using the fitness function to determine the most successful parents based on their accuracy in classification. Afterward, the crossover operation is carried out by selecting random parts of features from the two strongest parents and merging them to create a new offspring. Mutation is then applied by randomly adding or removing features through flipping selected bits in the child solution. Table 1 demonstrates an instance of mutation. Assuming our dataset has five features, and the resulting sequence after crossover is 11101. Genes represented by 1 indicate that a feature is included in subset, while genes represented by 0 indicate that a feature is not included. Upon examining the features, it is evident that Feature 4 is not included in the selection. If mutation were not performed, accuracy would be computed with the remaining four features. However, mutation involves randomly selecting and altering bits. As depicted in the table, the mutation produced the sequence 10101. As a result, Feature 2 will not be considered in model evaluation after the mutation.

After calculating feature importance with Shapley values, all features above a threshold are given as input to the genetic algorithm. By its nature, the genetic algorithm must have a stopping criterion. As the number of features to be included in the genetic algorithm process is reduced, features that can achieve higher success in a shorter time can be selected. Therefore, a filtering with shapley value is used in our model. The flow diagram of our hybrid feature selection approach using Shapley Values and Genetic Algorithm is shown in Figure 2.

Table 1. Mutation on features

Description	Feature 1	Feature 2	Feature 3	Feature 4	Feature 5
Before mutation	1	1	1	0	1
After mutation	1	0	1	0	1

Algorithm 1. Feature selection with Shapley values

```
Initialize an empty dictionary to store the Shapley values for each feature
For each subset of features 'S'
    Compute the marginal contribution of each feature 'i' in 'S':
    Select the features in 's' from input 'X' and split the remaining features into X_not_S
    Compute the model's predictions on both X_S and X_not_S
    Compute the difference between predictions with and without feature 'i' in 'S'
    Take the mean of the differences over all samples in 'X'
For each feature 'i'
    Shapley_value_of_i = Sum of the marginal contributions that contain 'i' / total number of subsets that contain 'i'
    Store Shapley_value_of_i in dictionary with key 'i'
```

Algorithm 2. Feature selection with Genetic Algorithm

```
Initialize population: generate a population where each individual is a binary feature vector with a length equal to the total number of features in dataset
Evaluate the fitness of the initial population
For each generation i=1 to maxGenerations
    Select parent for the next generation
    Create offspring using crossover and mutation
    Evaluate the fitness of the offspring
    Replace the least fit individuals with the offspring
Select the best individual as solution
```

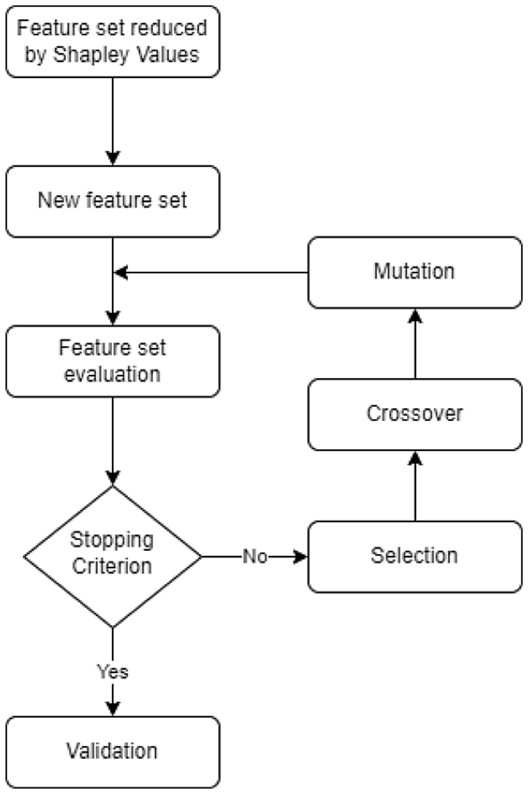

Figure 2. Flow diagram of feature selection step

Performance evaluation

Our automated preprocessing model has 3 steps: imputation, scaling, and feature selection. The imputation module of our automated preprocessing model is an important step in the data cleaning and preparation process. It is used to fill in missing data values in the dataset before further analysis is conducted. In this module, there are three different imputation methods available: mean imputation, median imputation, and most frequent imputation.

To implement the imputation module, the missing values in the dataset are first identified. Then, the selected imputation method is applied to replace the missing values with appropriate values based on the non-missing values in the dataset. This process ensures that there are no missing values in the dataset, which is essential for building a predictive model.

To evaluate the accuracy of the proposed model and approach, three datasets were used in this study. The first dataset, X-IIoTID [23], contains data on Industrial Internet of Things (IIoT) systems, which are widely used in industrial settings. The second dataset, KddCup99 [24], is a well-known dataset in the field of intrusion detection that contains network traffic data from a simulated military network. Finally, the third dataset, NSL-KDD [24], is a benchmark dataset that is commonly used to evaluate intrusion detection models.

By using these three diverse datasets, we can assess the robustness of the proposed model and approach across different settings and scenarios. This allows us to draw more general conclusions about the effectiveness of the proposed method and its potential for real-world applications.

Table 2 displays the results of applying various imputation techniques to the X-IIoTID, KddCup99, and NSL-KDD datasets. Our model applies these techniques one by one to dataset and records the accuracy achieved for each imputation method in the database. The parameter

Table 2. Results of imputation methods

Dataset	Imputation method	Accuracy
X-IIoTID	Mean	0.9967
X-IIoTID	Median	0.9972
X-IIoTID	Most frequent	0.9967
KddCup99	Mean	0.9998
KddCup99	Median	0.9998
KddCup99	Most frequent	0.9998
NSL-KDD	Mean	0.9994
NSL-KDD	Median	0.9994
NSL-KDD	Most frequent	0.9994

Table 3. Results of scaling methods

Dataset	Scaling method	Accuracy
X-IIoTID	Standard	0.9974
X-IIoTID	Min-max	0.9969
X-IIoTID	Robust	0.9970
KddCup99	Standard	0.9998
KddCup99	Min-max	0.9998
KddCup99	Robust	0.9998
NSL-KDD	Standard	0.9994
NSL-KDD	Min-max	0.9992
NSL-KDD	Robust	0.9994

information associated with the highest accuracy is then selected as input for the scaling step, which is the second step of our model. Table 2 shows that applying the median imputation method for missing values on X-IIoTID dataset, results in higher accuracy. The scaling step includes this imputation method as a parameter for handling missing values. Since there are no missing values in the KddCup99 and NSL-KDD datasets, no distinction could be observed through imputation methods. Therefore, the accuracy values on these datasets remained the same.

In the second step of our model, which is commonly referred to as the scaling step, our model utilize three distinct scaling methods to preprocess the data: the min-max scaler, standard scaler, and robust scaler. The min-max scaler scales the data to a fixed range, usually between 0 and 1, by subtracting the minimum value and dividing by the range of the data. The standard scaler, on the other hand, scales the data to have a mean of 0 and a standard deviation of 1. The robust scaler is similar to the standard scaler, but it is more robust to outliers because it uses median and interquartile range rather than the mean and standard deviation.

After applying each scaling method, we record the accuracies obtained in the database to compare the performance of each method. The scaling method with the highest accuracy from this step is then selected as a parameter for the last step of our model, the feature selection step. This step involves selecting the most important features that contribute to the prediction model, based on the scaling method used. By selecting the best scaling method for our data, we can improve the accuracy of our predictive model, thus increasing the overall performance of our system. Table 3 displays the accuracy results obtained from the scaling procedures employed by our model.

In our automated preprocessing model, the final step is feature selection, which utilizes the imputation and scaling techniques developed in the first two steps. While a genetic algorithm can be used alone for feature selection, it cannot determine how much each feature contributes to the overall accuracy. However, by randomly selecting features and analyzing the resulting accuracy, we can gain insights into their importance.

To enhance the accuracy and interpretability of our feature selection process, we employ a hybrid approach. We first use Shapley values to measure the contribution of each individual feature to the overall accuracy. We then remove any features that fall below a certain threshold value. This step helps us eliminate features that have little or no impact on the accuracy of our model. After identifying the relevant features, we use a genetic algorithm to create subsets of features that are likely to contribute the most to the accuracy of our model. By doing so, we ensure that our feature subsets are comprised only of the most impactful features.

The X-IIoTID dataset is a complex dataset that requires careful feature selection to achieve accurate results. Figure 3 displays a feature importance graph for this dataset, which provides insights into the impact of each feature on the overall accuracy. As the graph reveals, the bottom 6 features have little or no impact on the accuracy of the model.

Failing to perform a feature importance check could result in these 6 features being included in a feature subset in the genetic algorithm solutions. This would not only decrease the accuracy of the model but also increase the training and test times. To prevent this issue, our model incorporates a feature importance check before feeding the genetic algorithm. In the example shown in Figure 3, even if the threshold value is set to 0 for X-IIoTID dataset, the 6 features at the bottom of the graph will be eliminated. This emphasizes the

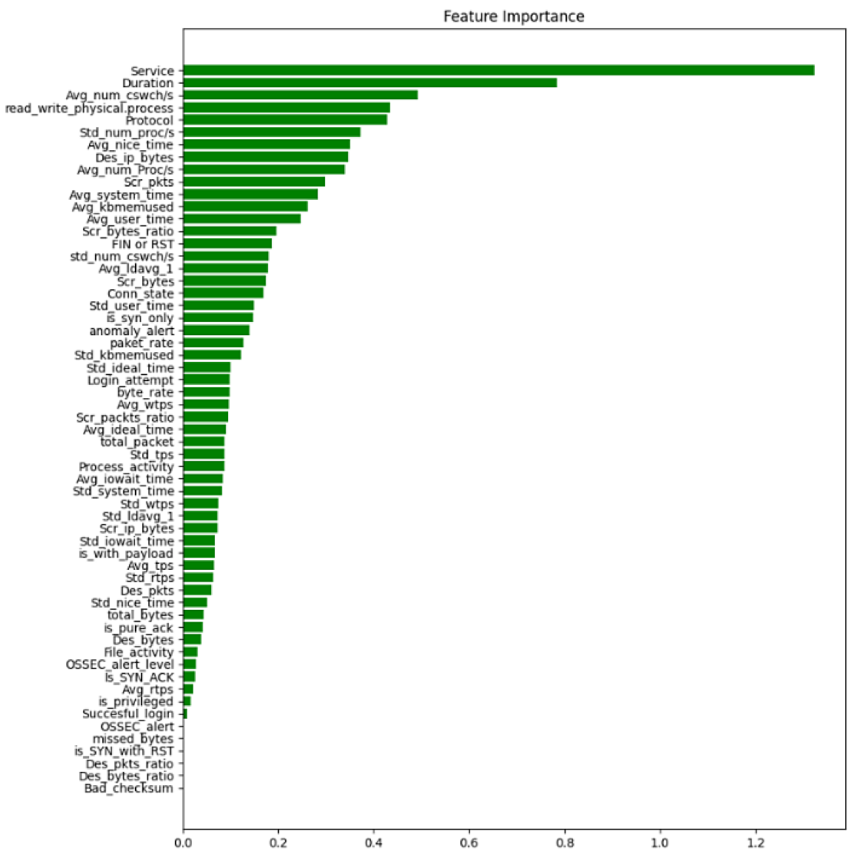

Figure 3. Feature importance with SHAP on X-IIoTID dataset

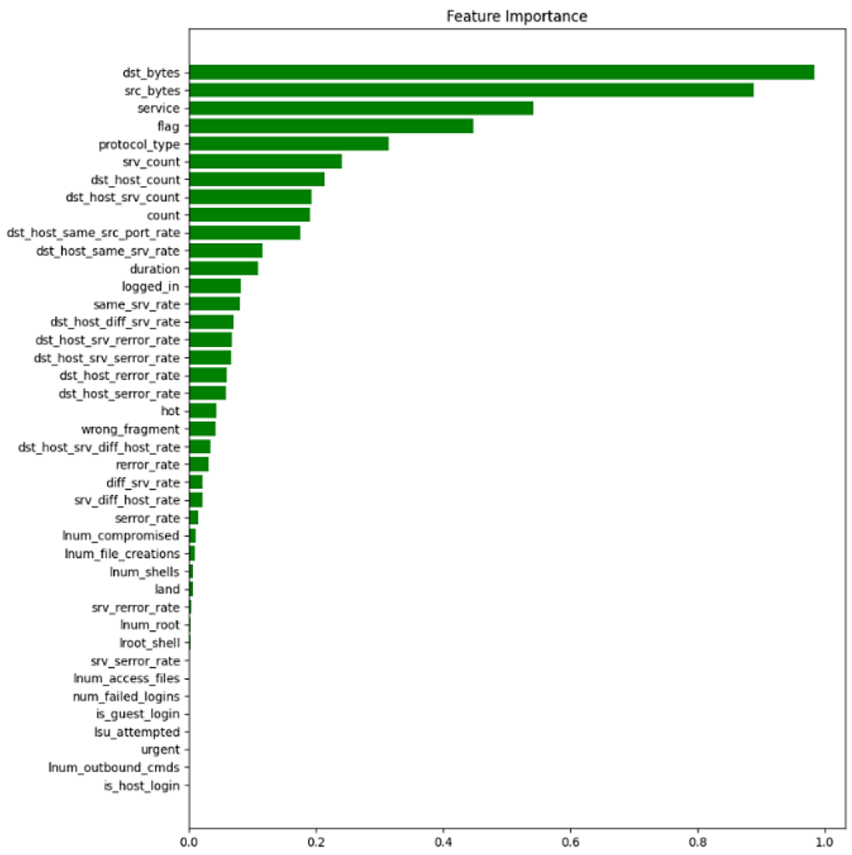

Figure 4. Feature importance with SHAP on X_KddCup99 dataset

importance of careful feature selection and how it can help streamline the training process and increase the accuracy of the model.

The feature importance graph for the X-IIoTID dataset plays a critical role in ensuring accurate feature selection. Our model uses this method to eliminate features that do not contribute to the overall accuracy, preventing the inclusion of unnecessary features in the feature subset. This results in a more efficient training process and a higher accuracy model. The results of our model's hybrid feature selection using shapley value and genetic algorithm are displayed in Table 4.

Feature importance graphs of KddCup 99 and NSL-KDD datasets are shown in Figure 4 and Figure 5, respectively. The figures show that certain features in both datasets have a feature importance value of 0, meaning they do not contribute to the prediction of the target variable. These features are considered irrelevant and may even add noise to the model. The algorithm starts by removing these unimportant features from the dataset, thus improving the model's performance and reducing the computational cost. By doing so, the hybrid approach can optimize the model's performance while keeping it computationally efficient.

RESULTS

Automated preprocessing is crucial for effective intrusion detection in IIoT systems. By using automated preprocessing techniques, it is possible to filter out irrelevant data, reduce the data volume, and transform the data into a format that is easy to analyze. This makes it easier to detect intrusions

Table 4. Results of feature selection with genetic algorithm and shapley values on X-IIoTID dataset

Feature No.	Accuracy
1, 3, 4, 5, 7, 8, 11, 14, 16, 18, 19, 20, 23, 27, 28, 29, 31, 34, 35, 36, 37, 39, 41, 43, 45, 47, 50, 52, 53, 54, 57, 58, 59	0.9972
1, 3, 4, 5, 7, 9, 10, 11, 14 ,15 ,16, 17, 18, 19, 20, 21, 25, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 41, 43, 45, 46, 47, 48,49, 50, 52, 54, 55, 58, 59	0.9973
1, 2, 3, 4, 5, 7, 8, 9, 11, 12, 14, 15, 16, 17, 20, 23, 28, 29, 30, 31, 32, 34, 35, 36, 38, 45, 47, 49, 52, 54, 55, 57, 58, 59	0.9975
2, 3, 4, 5, 7, 11, 14, 16, 17, 18, 21, 23, 25, 27, 29, 30, 31, 32, 33, 34, 36, 39, 41, 43, 44, 45, 48, 49, 50, 52, 53, 55, 57, 58, 59	0.9977
1, 2, 3, 4, 7, 11, 15, 16, 17, 18, 19, 21, 22, 25, 28, 29, 30, 33, 34, 38, 41, 43, 44, 48, 49, 50, 52, 53, 55, 57, 58, 59	0.9982

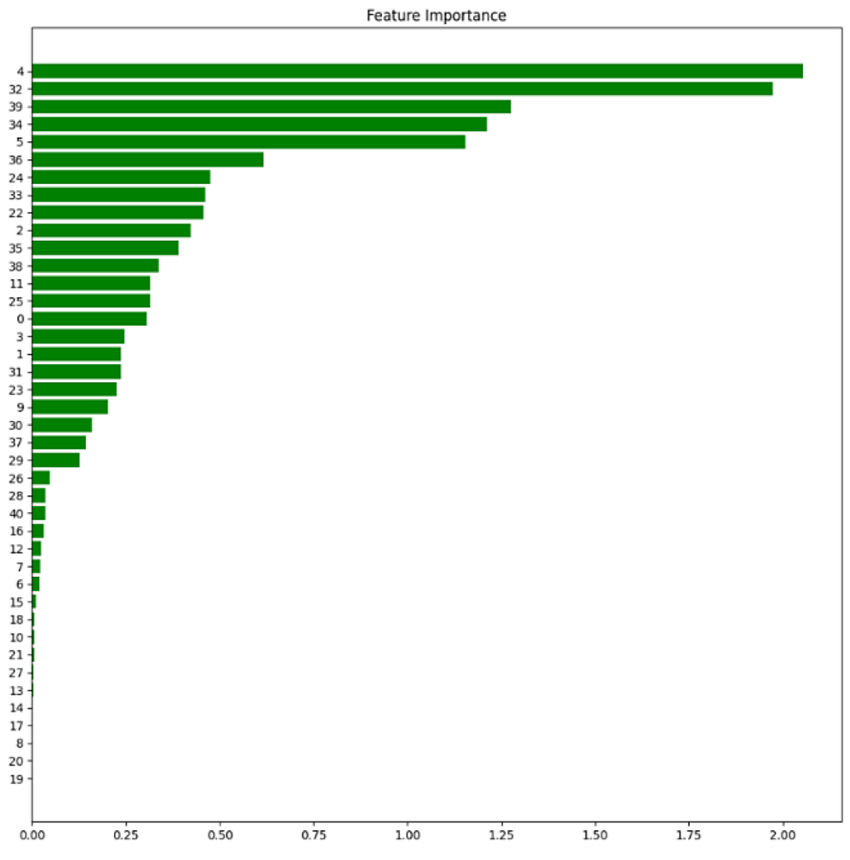

Figure 5. Feature importance with SHAP on X_NSL-KDD dataset

Table 5. Accuracy and elapsed time w/o preprocessing steps by using hybrid feature selection

Dataset	Accuracy (before FS)	Accuracy (after FS)	Elapsed time for training (before FS) (s)	Elapsed time for training (after FS) (s)	Number of removed features
X-IIoTID	0.9967	0.9982	194.67	143.29	27
KddCup99	0.9998	0.9999	477.93	328.44	19
NSL-KDD	0.9994	0.9999	17.3	12.1	16

Table 6. Accuracy and elapsed time w/o preprocessing steps by using genetic algorithm only

Dataset	Accuracy (before FS)	Accuracy (after FS)	Elapsed time for training (before FS) (s)	Elapsed time for training (after FS) (s)	Number of removed features
X-IIoTID	0.9967	0.9975	194.67	154.42	23
KddCup99	0.9998	0.9999	477.93	364.68	16
NSL-KDD	0.9994	0.9999	17.3	14.4	12

Table 7. Classification report of XGBoost

Specificattion	Precision	Recall	F1-Score	Support
Normal	0.9983	0.9981	0.9983	79729
Attack	0.9981	0.9982	0.9982	84247
Accuracy				0.9982

in real-time, improve system efficiency, and ensure the safety and security of IIoT systems.

Upon analyzing the data presented in Table 5, it becomes evident that there is a notable improvement in accuracy and reduction in training/test time after preprocessing the data.

Table 6 displays the outcomes of our automated preprocessing approach using genetic algorithm-based feature selection, both with and without feature selection. Table 5, on the other hand, employed Shapley values for initial filtering during feature selection. In Table 6, feature selection was carried out solely with genetic algorithm, without incorporating the feature importance information from Shapley values. The absence of feature importance information in the selection process resulted in a greater number of features, which in turn led to reduced accuracy and longer training/testing times.

Intrusion detection with machine learning is a classification task where the goal is to correctly classify network traffic or events as either normal or malicious. In this context, precision, recall, F1-score, and support are evaluation metrics used to measure the performance of a classification model.

- Precision – is the ratio of true positive (TP) instances to the total number of instances classified as positive. It measures the accuracy of the model when it predicts a positive class. A high precision score indicates that the model has fewer false positives and is good at correctly predicting positive instances.
- Recall – is the ratio of true positive (TP) instances to the total number of positive instances. It measures the ability of the model to correctly identify positive instances, i.e., its ability to detect malicious traffic. A high recall score indicates that the model has fewer false negatives and is good at detecting malicious traffic.
- F1-score – is the harmonic mean of precision and recall. It provides a balance between precision and recall and is a useful metric for comparing the performance of different models. A high F1-score indicates a good balance between precision and recall.
- Support – is the number of instances in the test set that belong to each class. It is useful for evaluating the model's ability to generalize to new data.

Our automated preprocessing model with XGBoost was used to preprocess the X-IIoTID dataset, and the resulting classification report is presented in Table 7.

CONCLUSIONS

This research proposes a comprehensive hybrid feature selection approach for intrusion detection in Industrial Internet of Things by integrating multiple steps, including imputation, scaling, and feature selection using Shapley values and genetic algorithm-based automated preprocessing. The proposed approach overcomes the challenges associated with detecting malicious activities in large and complex IIoT systems by identifying the most relevant features for intrusion detection. Our experimental results demonstrate that the proposed approach significantly improves the performance of intrusion detection systems in terms of accuracy, precision, recall, and F1-score. Moreover, the results demonstrate that the proposed approach can reduce the number of features needed for intrusion detection, which in turn reduces the computational and communication overhead of the system. The proposed approach can be used as an effective tool for improving the security of IIoT systems by detecting malicious activities accurately and efficiently. Future research can focus on extending the proposed approach to handle more complex scenarios in the IIoT environment and explore the potential of incorporating other machine learning techniques to improve the performance of the proposed approach. Our proposed automated preprocessing approach and hybrid feature selection for intrusion detection in Industrial Internet of Things has shown promising results.

Acknowledgements

This study is a part of the Ph.D. thesis at Atatürk Strategic Studies and Graduate Institute, National Defence University, Istanbul, Turkey. We would like to thank Siemens Turkey for their support in the completion of the study.

REFERENCES

1. R. Vinayakumar, M. Alazab, K.P. Soman, P. Poornachandran, A. Al-Nemrat, S. Venkatraman, Deep Learning Approach for Intelligent Intrusion Detection System, IEEE Access, 7, 41525-41550, 2019.
2. D.P. Hostiadi, Y.P. Atmojo, R.R. Huizen, I.M.D. Susila, G.A. Pradipta, I.M. Liandana, A New Approach Feature Selection for Intrusion Detection System Using Correlation Analysis, International Conference on Cybernetics and Intelligent System (ICORIS), Prapat, Indonesia, 2022.
3. V. Ravindranath, S. Ramasamy, R. Somula, K.S. Sahoo, A.H. Gandomi, Swarm Intelligence Based Feature Selection for Intrusion and Detection System in Cloud Infrastructure, IEEE Congress on Evolutionary Computation (CEC), Glasgow, UK, 2020.
4. T. Janarthanan and S. Zargari, Feature selection in UNSW-NB15 and KDDCUP'99 datasets, International Symposium on Industrial Electronics (ISIE), Edinburgh, UK, 2017.
5. A. Saber, M. Abbas, B. Fergani, Two-dimensional Intrusion Detection System: A New Feature Selection Technique, International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH), Boumerdes, Algeria, 2021.
6. N. Sampath, M.A. Jerlin, L.B. Kritkika, A. Anitha, Intrusion Detection in Software Defined Networking using Genetic Algorithm, International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, 2020.
7. O.D. Okey, D.C. Melgarejo, M. Saadi, R.L. Rosa, J.H. Kleinschmidt, D.Z. Rodriguez, Transfer Learning Approach to IDS on Cloud IoT Devices Using Optimized CNN, IEEE Access, 11, 1023-1038, 2023.
8. H. Aiods and P. Tomás, Neighborhood-aware autoencoder for missing value imputation, European Signal Processing Conference (EUSIPCO), Amsterdam, Netherlands, 2020.
9. C. Yin, Y. Zhu, J. Fei i X. He, An Implementation of Intrusion Detection System Using Genetic Algorithm, IEEE Access, 5, 21954-21961, 2017.
10. M. Zolanvari, M.A. Teixeira, L. Gupta, K.M. Khan, R. Jain, Machine Learning-Based Network Vulnerability Analysis of Industrial Internet of Things, IEEE Internet of Things Journal, 6(4), 6822-6834, 2019.
11. V.D. Katkar and D.S. Bhatia, Lightweight approach for detection of denial of service attacks using numeric to binary preprocessing, International Conference on Circuits, Systems, Communication and Information Technology Applications (CSCITA), Mumbai, India, 2014.
12. A. Telikani, J. Shen, J. Yang, P. Wang, Industrial IoT Intrusion Detection via Evolutionary Cost-Sensitive Learning and Fog Computing, IEEE Internet of Things Journal, 9(2), 23260-23271, 2022.
13. G.A. Da Silva Oliveira, P.S. Silva Lima, F. Kon, R. Terada, A stacked ensemble classifier for an intrusion detection system in the edge of IoT and IIoT Networks, IEEE Latin-American Conference on Communications (LATINCOM), Rio de Janeiro, Brazil, 2022.
14. A. Khacha, R. Saadouni, Y. Harbi, Z. Aliouat, Hybrid Deep Learning-based Intrusion Detection System for Industrial Internet of Things, International Symposium on Informatics and its Applications (ISIA), M'sila, Algeria, 2022.
15. A. Zainudin, R. Akter, D.S. Kim, J.M. Lee, Towards Lightweight Intrusion Identification in SDN-based Industrial Cyber-Physical Systems, Asia Pacific Conference on Communications (APCC), Jeju Island, Korea, Republic of, 2022.
16. H. Aidos and P. Tomás, Neighborhood-aware au-

toencoder for missing value imputation, European Signal Processing Conference (EUSIPCO), Amsterdam, Netherlands, 2021.

17. A. Heryanto, D. Stiawan, M.Y. Bin Idris, M.R. Bahari, A.A. Hafizin, R. Budiarto, Cyberattack feature selection using correlation-based feature selection method in an intrusion detection system, International Conference on Electrical Engineering, Computer Science and Informatics (EECSI), Jakarta, Indonesia, 2022.
18. M. Al-Hawawreh, E. Sitnikova, N. Aboutorab, X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial internet of things, IEEE Internet of Things Journal, 9(5), 3962-3977, 2022.
19. M. Tavallaee, E. Bagheri, W. Lu, A.A. Ghorbani, A detailed analysis of the KDD CUP 99 data set, IEEE Symposium on Computational Intelligence for Security and Defense Applications, Ottawa, ON, Canada, 2009.
20. P.B. Udas, E. Karim, K.S. Roy, SPIDER: A shallow PCA based network intrusion detection system with enhanced recurrent neural networks, Journal of King Saud University - Computer and Information Sciences, 34(10), 10246-10272, 2022.
21. J. Ahmad, S.A. Shah, S. Latif, F. Ahmed, Z. Zou, N. Pitropakis, DRaNN_PSO: A deep random neural network with particle swarm optimization for intrusion detection in the industrial internet of things, Journal of King Saud University - Computer and Information Sciences, 34(10), 8112-8121, 2022.
22. M. Mazini, B. Shirazi, I. Mahdavi, Anomaly network-based intrusion detection system using a reliable hybrid artificial bee colony and AdaBoost algorithms, Journal of King Saud University - Computer and Information Sciences, 31(4), 541-553, 2019.
23. S.T. Ikram and A.K. Cherukuri, Intrusion detection model using fusion of chi-square feature selection and multi class SVM, Journal of King Saud University – Computer and Information Sciences, 29(4), 462-472, 2017.
24. Almutairi YS, Alhazmi B, Munshi AA. Network intrusion detection using machine learning techniques. Advances in Science and Technology Research Journal. 2022; 16(3): 193-206.