Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(8): 13824–13848.
DOI: 10.3934/mbe.2023615
Received: 08 April 2023
Revised: 20 May 2023
Accepted: 29 May 2023
Published: 16 June 2023

Research article

Multi-step attack detection in industrial networks using a hybrid deep learning architecture

Muhammad Hassan Jamal[1], **Muazzam A Khan**[1,2], **Safi Ullah**[1], **Mohammed S. Alshehri**[3], **Sultan Almakdi**[3], **Umer Rashid**[1], **Abdulwahab Alazeb**[3] **and Jawad Ahmad**[4,*]

[1] Department of Computer Sciences, Quaid-i-Azam University, Islamabad 45320, Pakistan

[2] ICESCO Chair Big Data Analytics and Edge Computing, Quaid-i-Azam University, Islamabad 45320, Pakistan

[3] Department of Computer Science, College of Computer Science and Information Systems, Najran University, Najran 61441, Saudi Arabia

[4] School of Computing, Engineering and the Built Environment, Edinburgh Napier University, EH10 5DT, Edinburgh, UK

* **Correspondence:** Email: J.Ahmad@napier.ac.uk.

Abstract: In recent years, the industrial network has seen a number of high-impact attacks. To counter these threats, several security systems have been implemented to detect attacks on industrial networks. However, these systems solely address issues once they have already transpired and do not proactively prevent them from occurring in the first place. The identification of malicious attacks is crucial for industrial networks, as these attacks can lead to system malfunctions, network disruptions, data corruption, and the theft of sensitive information. To ensure the effectiveness of detection in industrial networks, which necessitate continuous operation and undergo changes over time, intrusion detection algorithms should possess the capability to automatically adapt to these changes. Several researchers have focused on the automatic detection of these attacks, in which deep learning (DL) and machine learning algorithms play a prominent role. This study proposes a hybrid model that combines two DL algorithms, namely convolutional neural networks (CNN) and deep belief networks (DBN), for intrusion detection in industrial networks. To evaluate the effectiveness of the proposed model, we utilized the Multi-Step Cyber Attack (MSCAD) dataset and employed various evaluation metrics.

Keywords: deep learning; intrusion detection; industrial networks; multi-step attacks; convolutional neural networks

1. Introduction

A network can be seen as a tangible manifestation of interconnected components. When comparing industrial networks to social networks, communication networks, or electrical networks, it becomes apparent that the entities involved in industrial networks are active participants in economic processes. These networks play a vital role in converting raw materials into the end products and services that are ultimately consumed by the public [1, 2]. Consequently, the relationships between actors within industrial networks are often conceptualized through financial transactions that occur within the framework of long-term partnerships. These interconnected links are what make industrial networks viable and successful [3, 4]. Typically, networks are utilized for data transfer, and certain networks are better suited for handling specific data volumes. Industrial networks enable the connection of various devices over long distances, facilitating communication between them by efficiently transmitting and receiving substantial volumes of data [5]. Therefore, the number of internet-connected devices in industrial settings has experienced a recent increase, primarily driven by emerging technologies in Internet of Things (IoT) networks [6]. While these technologies make administrative tasks easier, they also face the challenge of cyber attacks. One such system commonly used in industrial environments is Supervisory Control and Data Acquisition (SCADA) [7]. Sodinokibi, an affiliate of REvil, launched a Stuxnet [8] attack against Acer, while the Darkside cyber-criminal organization launched an attack against Colonial Pipeline [9].

Recently, hackers have been targeting not just traditional commercial networks but also industrial networks [10]. Twenty-one lines of code were demonstrated by the US government in 2007 as a potential means of damaging a power plant generator [11]. To render an out-of-phase generator, unless the attackers connected and disconnected it from the grid on a regular basis [12]. More than 80,000 people in Ukraine were left without electricity after cyber-criminals hijacked the networks of two power distribution companies in late 2015 and early 2016 [13]. Hackers got into a German steel firm in 2015 and tampered with a blast furnace, severely damaging the plant [14]. An attack scenario is illustrated in Figure 1. These attack scenarios show that traditional systems cannot be adequately protected using standard security methods. Consequently, cyber security is increasingly becoming an inherent component of today's industrial networks [15].

Numerous security solutions, such as intrusion detection systems (IDS) based on machine learning (ML) and deep learning (DL) models [16], have been proposed by several research studies to protect industrial networks from both known and unknown (zero-day) attacks [17]. An IDS is software that monitors the network for any suspicious activities, like a failed network intrusion attempt or the use of a hacked account [18]. An IDS is considered to be the most effective form of security because it is designed to identify intrusions by spotting unusual anomalies in sensor data or actuator activity [19]. Hence, an IDS plays a crucial role in preserving the security of industrial networks [20].

There are a number of problems with the current batch of IDSs [21], such as poor detection rates, excessive false positives, and sluggish real-time functionality during attempted impersonations [22]. Due to insufficient security solutions, the attacker breaks the firewall to attack the networks, which results in a huge loss of confidential data of the industrial networks. They either sneak into the network and snitch the sensitive information without making any changes, or they obtain unauthorized access and modify the confidential data by encrypting, deleting, or tampering with it, thus, badly damaging the industrial network. The detection of malicious attacks in industrial networks is crucial, as these attacks can result in system failures, network disruptions [23], data damage, or the theft of confidential information [24]. Thus, enhancing the security of industrial networks necessitates the creation of reliable and efficient systems for detecting anomalies, based on continuous monitoring of the actual state of the system.

To automatically detect malicious attacks in real-time, DL algorithms exhibit a superior performance compared to traditional ML algorithms, particularly when handling large amounts of data [25]. DL is highly effective in detecting zero-day attacks and achieving a high detection rate [26], as it has the capability to automatically identify correlations within the data [27]. Therefore, in this study, DL algorithms convolutional neural networks (CNN) and deep belief networks (DBN) are used to develop an IDS which specifically meets the requirements of industrial network security. To improve detection rates while lowering training and generalization errors, we have proposed an integrated intrusion detection model that comprises three layers namely CNN, DENSE, and DBN. The reason for choosing CNN and DBN for IDS is due to their superior performance compared to other ML and DL methods [28]. The results show that the proposed model offers an optimal performance within a shorter time frame when compared to other models.

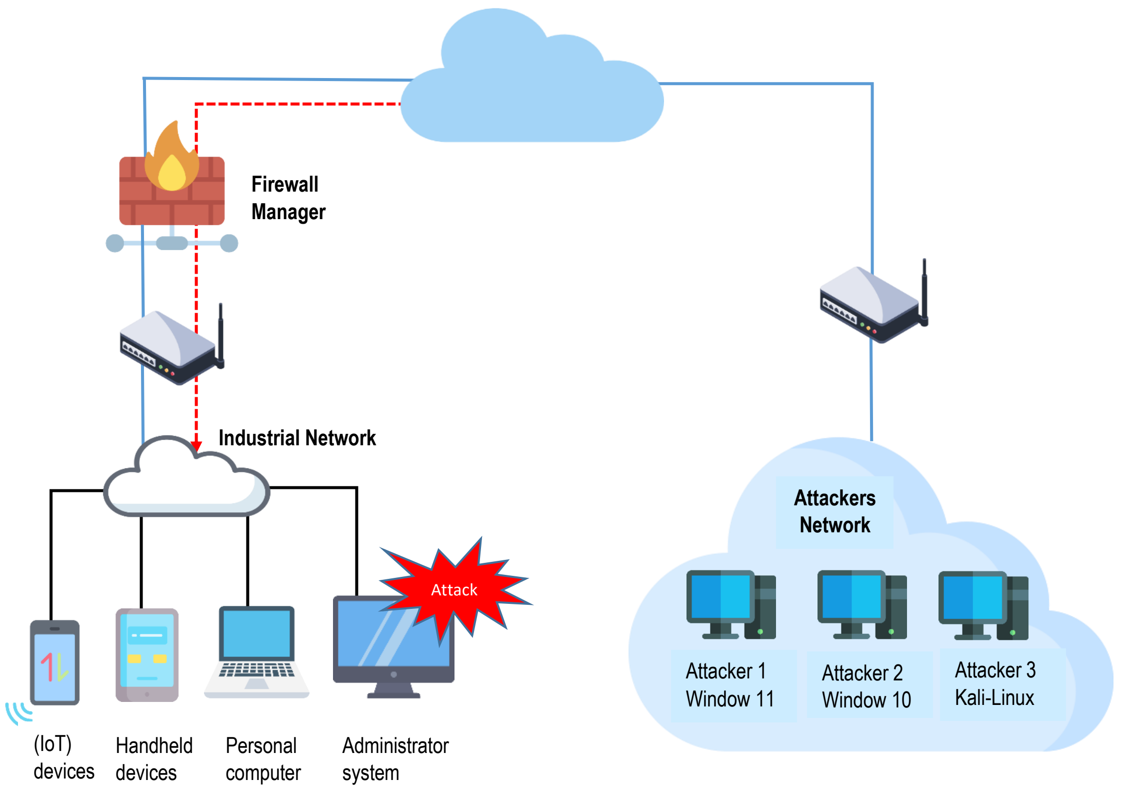

Figure 1. An attack scenario on a device in the network.

2. Related work

2.1. Background

With the rise of interconnected networks, industrial networks are now relying on the vast amount of information available online. However, increasing integration has left industrial network security exposed to a variety of cyberattacks [29]. As such, it is imperative to integrate measures to detect these potential attacks and establish defense mechanisms [30]. One of the most effective solutions is the use of IDSs.

These systems keep an eye out for any suspicious behavior on a network and can spot intrusions by seeing potentially harmful abnormalities in sensor data or actuator activity [19]. The primary goal of this research study is to develop an improved model for IDSs that provides optimal accuracy rates and enhances the performance of evaluation measures used in experiments [31].

2.2. State of the art

The use of ML and time series-based anomaly detection methods in [32] helps in monitoring industrial operations data present in networks for signs of cyber-attacks. The data analyzed include Modbus-based gas pipeline control traffic and OPC UA-based batch processing traffic. The resulting accuracy of 90.8% is slightly worse than before. The F1-score is doing quite well with a 94.9% recall. The algorithms still work well, despite a large amount of missing data in this dataset. In actual use, this will allow attacks on industrial networks to go undetected, which is a major security risk. To improve safety, more resources are needed.

The knowledge distillation triplet convolution neural network (KD-TCNN) model was developed to enhance the performance of IDS in industrial CPS. The proposed model reduces the computational cost and size by 86% while maintaining a 0.4% drop in accuracy compared to the current model. To process and analyze huge volumes of data and comprehend the relationships between categories in the teacher network, knowledge distillation aims to make the student network's outputs identical to those of the teacher network. To perform an efficient detection of intrusion, the model must undergo two stages of training: first, the preliminary training of the teacher model, and second, the training of the KD-TCNN model. To bridge the gap between similar inputs for knowledge distillation, deep metric learning is applied, which improves the performance of the student model [33].

The data from the Integrated Automation lab's process control plant was utilized to gather metrics pertaining to ICS. CNNs were employed to assess the efficiency of detecting injection attacks. A three-layer CNN architecture with rectified linear activation unit (ReLU) activation and a SoftMax classification layer was constructed to distinguish between benign and malicious data. The proposed CNN model's efficacy was gauged through a range of metrics such as F1-score, accuracy, recall, precision, and Cohen's kappa coefficient. In terms of performance, CNN demonstrated a superiority over other deep learning algorithms. However, the outcome depends on the program and data utilized. By combining various DL and ML techniques, the effectiveness of the security mechanism can be further improved. Alternative DL techniques such as recurrent neural networks (RNN) can also be employed [34].

The intrusion detection system based on CNN has the capability to identify different attack types, which were evaluated using datasets such as Network Security Laboratory-Knowledge Discovery and

Data Mining (NSL-KDD) and University of New South Wales-Network Behavior 15 (UNSW-NB 15). Performance indicators such as recall, F1-score, and precision, were calculated and contrasted to those of other DL techniques. A CNN has been tested on the NSL-KDD and UNSW-NB 15 datasets for intrusion detection, resulting in an improved detection accuracy when compared to previous CNN-based methods. However, when it came to classifying attacks with multiple classes, the CNN did not perform better than other DL-based IDSs, such as SAE and DBN. For a more specific and accurate development, creating a dataset within the ICS context should be considered. MATLAB can be used to capture real-time network traffic data and DL algorithms can then be applied to this network for enhanced results [35].

The use of a Stacked Auto-encoder (SAE) combined with a CNN can reduce high-dimensional data to generalizations and essential characteristics that describe the machine tool's operating state. The process begins with the use of an unsupervised SAE for generic feature extraction from industrial process data. The proposed model's effectiveness in the updated production line is then validated by fine-tuning a one-dimensional CNN classifier with supervised data. It is evident from the experimental results that the proposed model is a potent and versatile solution for monitoring machine tool production data, minimizing the complexity features with high-dimensional, and having positive economic impacts on the industry [36].

A proposed model for SCADA [37] network data detection has been suggested as a means of creating a secure architecture for ICS networks. The proposed architecture creates two ensemble-based detection methods by combining a DBN with a conventional classifier, such as a support vector machine (SVM). The results of the experiments support the effectiveness of the proposed DBN ensemble approach in addressing the DBN structure selection problem and providing dependable attack detection for the secure operation of SCADA systems. However, the approach has some limitations, including using a neural network (NN)-based classifier within the DBN framework. Although DBNs are effective in extracting features from raw data, the NN-based classifiers they use can limit their pattern recognition precision. Additionally, the proposed training method for the DBN is not designed for real-time situations [38].

A proposed method for diagnosing intrusions in industrial robots, based on joint DBN information fusion technology, addresses the limitations of the traditional fault diagnosis model. These limitations include low precision, inefficiency, instability, and a slow time-to-diagnosis in cases of multiple faults. To test the method, the researchers chose an industrial robot with a problem in one of its joint bearings. The results showed that the developed fault diagnostic approach is effective, with a test set accuracy of 97.96%. The proposed method has several advantages over the traditional diagnostic model, including a faster diagnosis time and an improved diagnosis efficiency, making it better suited for diagnosing multiple faults [39].

The detection of cyberattacks on SCADA-based industrial control systems (ICS) is addressed in a study that introduces the first-ever population extremal optimization (PEO)-based DBN network detection technique (PEO-DBN). To further improve the detection performance, a new ensemble learning strategy, called EnPEO-DBN, is proposed to combine the PEO-DBN approach. The simulation results depicted that the proposed PEO-DBN and EnPEO-DBN have superior performance compared to other methods and can be considered promising solutions for detecting cyberattacks on SCADA-based ICS. To speed up the fitness evaluation process, future work can take advantage of surrogate-assisted models, such as a Gaussian process regression or a Bayesian optimization [40].

As a means of securing IoT networks, [41] presented a hybrid DBN cyber intrusion detection system. In order to construct better attack detectors for network traffic, researchers have studied and addressed DBN's drawbacks. The results from the analysis are then integrated into a SoftMax Regression model to improve the security and accuracy of detection. The hybrid DBN model was trained and evaluated based on the original and unaltered dataset generated by the system. In terms of detecting and classifying intrusions, the suggested hybrid DBN model had a 99.72 % success rate. According to these findings, the model outperformed the current IDS. Further, the hybrid model offered a roughly 5% greater accuracy improvements compared to pre-hybrid DBN-based systems.

An interesting approach is used as a solution that aims to combine the strengths of two popular machine learning techniques: Long short-term memory (LSTM) and DBNs. By using LSTM to enhance the representation power of traditional shallow machine learning, and DBNs to extract non-linear components from the data, the researchers hope to achieve both high accuracy and high processing speed. The results of their approach appear to be promising, with recall rates as high as 97.3% and area under the curve (AUC) as high as 0.927, suggesting that this combination of techniques could be effective for solving complex problems in machine learning. However, as mentioned, this area of research is still being explored, and more work needs to be done to fully understand the trade-offs between precision, speed, and other factors when using this hybrid approach [42].

2.3. Existing datasets

The KDD-98 dataset, produced by the "Defense Advanced Research Projects Agency's Knowledge Discovery and Data Mining program" in 1998, was the first publicly available intrusion detection dataset, though it contained flaws like duplicate records [43]. NSL-KDD, introduced in 2009, improved on KDD-98, though KDD-99 still had issues like outdated attack methods and inaccurate network parameters [43]. The Cooperative Association for Internet Data Analysis (CAIDA) dataset, introduced in 2007 had limited information on DDoS attacks [44]. The University of Brescia (UNIBS) database (2009) by Gringoli et al. focused on profiling popular web apps but was unsuitable for detecting anomalies [45]. The Information Security Centre of Excellence (ISCX) 2012 dataset by Shiravi et al. had two network configurations (α and β) but lacked Hypertext Transfer Protocol Secure (HTTPS) protocol traffic [46]. DDoS 2016, released in 2016 by Alkassasbeh et al., was not useful for multi-step attack detection. The ADFA Linux (ADFA-LD) and ADFA Windows Datasets (ADFA-WD) were created using system call traces, but attackers can hide their tracks. Canadian Institute for Cybersecurity Intrusion Detection System (CIC-IDS) 2017 had 80 network parameters from a five-day simulated test but had missing values and no information on multi-stage attacks. TUIDS, developed by Tezpur University, used a hybrid of packet and flow formats but couldn't simulate multi-step attacks.

The new Multi-Step Cyber Attack (MSCAD) [47] dataset overcomes the limitations of prior methods by serving as a new benchmark for IDS. The MSCAD stands out with its ability to detect complex, multi-stage attacks, which are carried out through a variety of techniques such as volume-based DDoS, site crawling, and network-based DDoS. This dataset is built on a full network infrastructure and contains no duplicates or missing data, making it ready for use in training IDS systems without any further cleaning [48]. The attributes of the MSCAD have been compared to those in the literature, including the dataset used, evaluation measures, k-fold validation, balancing strategy, and response time, as presented in Table 1.

Table 1. Comparison of previous literature.

Author	Dataset	Evaluation measure	K-Fold validation	Balancing strategy	Response time
[32]	Modbus, OPC-UA	Accuracy, Precision, Recall, F-1 score	No	No	High
[38]	SCADA network	Accuracy, Precision	Yes	No	Medium
[39]	Industrial robot	Accuracy	No	No	Low
[40]	SCADA network	Accuracy	No	No	Medium
[35]	NSL-KDD, UNSW-NB15	Accuracy, Precision, Recall, F-1 score	Yes	No	Medium
[34]	PCP from IAE	Accuracy, Precision, Recall, F-1 score	No	No	High
[41]	SCADA network	Accuracy, F-1 score	No	No	Medium
[33]	NSL-KDD, CIC-IDS 2017	Accuracy, Precision, F-1 score	Yes	Yes	High
[42]	NSL-KDD	Accuracy, Precision, Recall, F-1 score	No	No	Medium
[36]	Collected data from industry production line	Accuracy	No	No	Medium
Proposed Model	Multi-step cyber attack dataset	Accuracy, Precision, Recall, F-1, G-mean score	Yes	Yes	Low

3. Proposed methodology

The following section comes up with a comprehensive overview of the proposed methodology. The data has undergone pre-processing and has been trained and tested. The diagram in Figure 2 showcases the general structure and flow of the model. The implementation involves the use of two DL algorithms, with a thorough explanation of how the CNN and DBN components are integrated into the proposed scheme.

3.1. Dataset

There are several public network datasets available for intrusion detection, such as KDD-98, NSL-KDD [43], CAIDA [44], UNIBS [45], ISCX 2012 [46], DDos 2016, ADFA-LD and ADFA-WD, CIC-ID 2017, and TUIDS. This experiment makes use of the MSCAD [20] dataset for industrial networks, as it is the latest dataset available and features a variety of attack scenarios that are well-suited for detecting multi-step attacks, as demonstrated in Table 2.

Table 2. Multi-step cyber attack dataset.

Classe	Encoding	Records
Brute force	0	88,502
HTTP DDoS	1	641
ICMP flood	2	45
Normal	3	28,501
Port scan	4	11,081
Web crawler	5	28

3.2. Data processing

3.2.1. Cleaning

Before analyzing a data set, it must undergo data cleaning, which consists of correcting or removing any incorrect, redundant, or otherwise undesired information. Data cleansing aims to remove ambiguity and other issues from your data. We must have clean data (i.e., information that our costly data analysis tools can process [49]).

Each dataset consists of many entries. Before training a model, inspecting a dataset for missing and undefined entries is essential. Training the model will produce errors if any data record includes an unexpectedly blank or undefined value. We should ensure that the dataset does not have any confusing data before training a model by cleaning it first. Several approaches [50, 51] are available for removing redundant or unclear entries from data sets. The dataset was cleansed using Python libraries (Pandas and NumPy) [52] and functions that provide true or false Boolean values to test for missing and enduring values. If this is true, then some of the values in the dataset are missing or infinite; however, if it is false, then the cleaning of the dataset is completed. The datasets were cleaned by converting all undefined records into blank spaces. Once the undefined values were transformed, the dataset was further refined by removing all empty entries and replacing any remaining undefined values with empty ones.

3.2.2. Numericalization

To achieve numericalization, one must first construct a string-to-integer encoding system and then individually apply it to each string. Typically, the attributes or labels of the datasets we work with, are available throughout multiple columns. These labels can be composed of either words or numbers. Typically, training data is labeled with either a phrase that makes it understandable or in a human-readable format. The process of label encoding [53] involves transforming human-readable labels into a format that can be processed by computers. This allows machine learning [54] algorithms to make more informed decisions based on these labels and is an essential step in the preparation of structured datasets for supervised learning. The label encoder searches for labels ranging from zero to n-1 and gives numerical values to these labels. This experiment quantifies categorical attributes using the label encoder method.

3.2.3. Normalization

During machine learning model training, data normalization reduces the impact of feature scales. Thus, our model can converge to accurate weights, therefore enhancing its accuracy. Normalization increases the model's capacity to predict outputs by bringing the features into a condition of homogeneity. The process of rescaling a numeric attribute with a real value to the interval 0 to 1 is what normalization means. This decreases the effect of the feature size on the training procedure [55]. Consequently, this leads to improved coefficients following the training. Normalization is a technique applied to datasets with varying value ranges, such as when one feature has values ranging from 0 to 1 and another has values ranging from 100 to 1000. This helps to equalize the data and makes it easier for the machine-learning model to be trained effectively.

The selected dataset in this experiment has features with varied range values; for example, some features have extremely high range values while others have extremely low range values. If we train the model using these variables, it will fail. This problem can be addressed by employing the normalization approach, which can be done by this equation to transform all non-zero values into one.

$$X_{norm} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.1}$$

3.2.4. Balancing strategies

The experiment's model is based on a dataset with a substantial number of records. Prior to training and testing the model, it's critical to guarantee that each class has the same amount of examples. This can be achieved through the use of balancing techniques such as oversampling and undersampling. Oversampling raises the minority class's percentage to the majority class, whereas undersampling decreases the majority class's share to match the minority class. However, oversampling can lead to overfitting because it involves repetitive data. The dataset used for this experiment was imbalanced, so we applied two balancing techniques, the Synthetic Minority Oversampling Technique (SMOTE) [56] and SMOTETomek, to ensure that the dataset was representative of the entire population.

SMOTE generates synthetic information using a k-nearest neighbor algorithm. The first phase of SMOTE involves randomly selecting data from the minority class and then determining those data's k-nearest neighbors. Then, the random data and the k-nearest neighbor are utilized to generate synthetic data. There is a version of the SMOTE called the Borderline-SMOTE. The name implies a connection to borders, which is indeed the case. In Borderline-SMOTE, synthetic data are generated only along the decision boundary between the two classes, whereas in SMOTE, they are generated randomly between the two classes [57]. The SMOTEENN and SMOTETomek sampling methods include oversampling (SMOTE) and undersampling (ENN, Tomek). SMOTE generates synthetic class samples to establish a statistical parity, whilst the latter is used to clean redundant information at the boundary of two classes to increase separation.

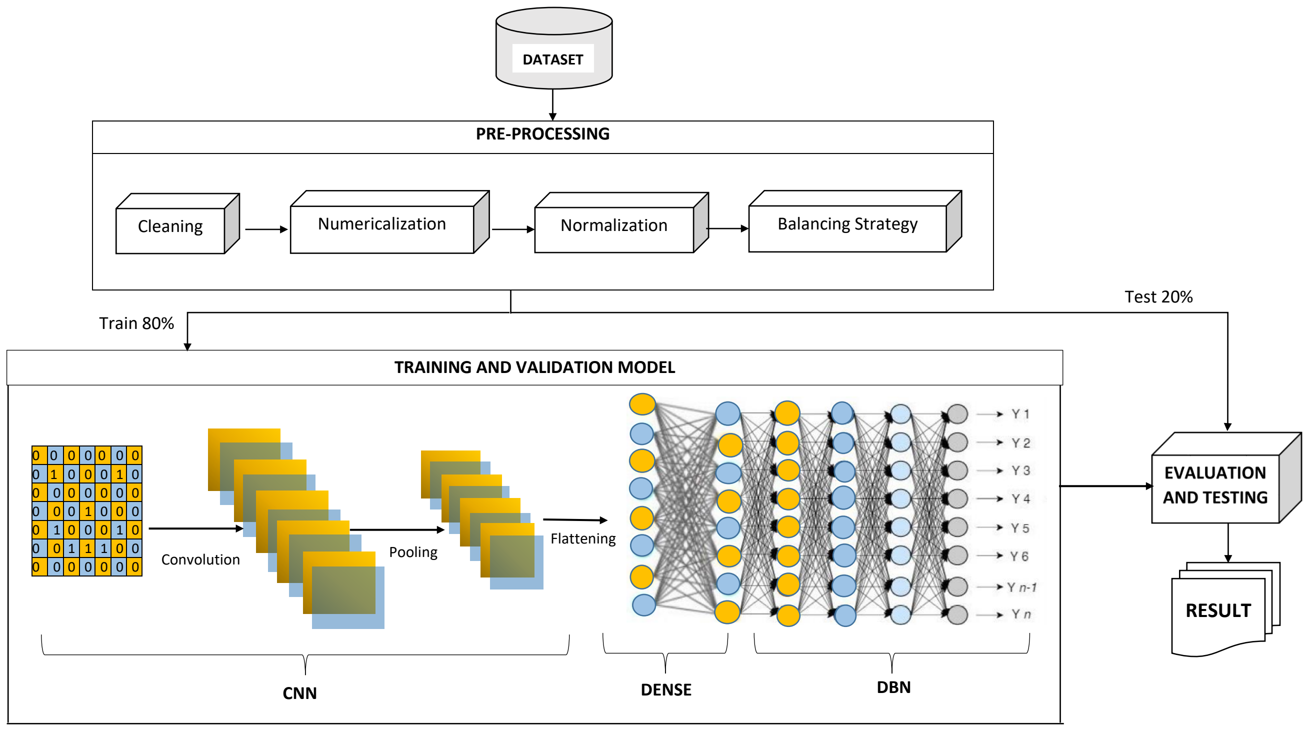

Figure 2. The Proposed architecture for CNN, DENSE, and DBN.

3.3. *Proposed model*

We have proposed an integrated intrusion detection model that is a combination of CNN and DBN. The proposed model contains three layers: CNN, DENSE, and DBN, which are depicted in Figure 2. All of the models utilized in the proposed model's layers are extensively explained for understanding. The model enhanced the performance for the detection of attacks [58] and reduced the response and training time in industrial networks.

3.3.1. CNN

Convolutional neural networks (CNNs) are a powerful tool for discovering features, as they reduce the need for human input [59]. The primary objective of CNN is to learn the most significant features of the input data [60]. A CNN is comprised of pooling layers, convolutional layers, and fully connected (FC) layers [61], which can be stacked to form a CNN architecture as illustrated in Figure 3. These layers also incorporate an activation function, which is a crucial aspect of their function [62]. The convolutional layer performs a mathematical operation, known as the convolution, between the input and a filter of size $M \times M$ to extract features from the input data. As the filter is moved over the input, the dot product of its $M \times M$ dimensions and the input is calculated, and the output is forwarded to the succeeding layer [63].

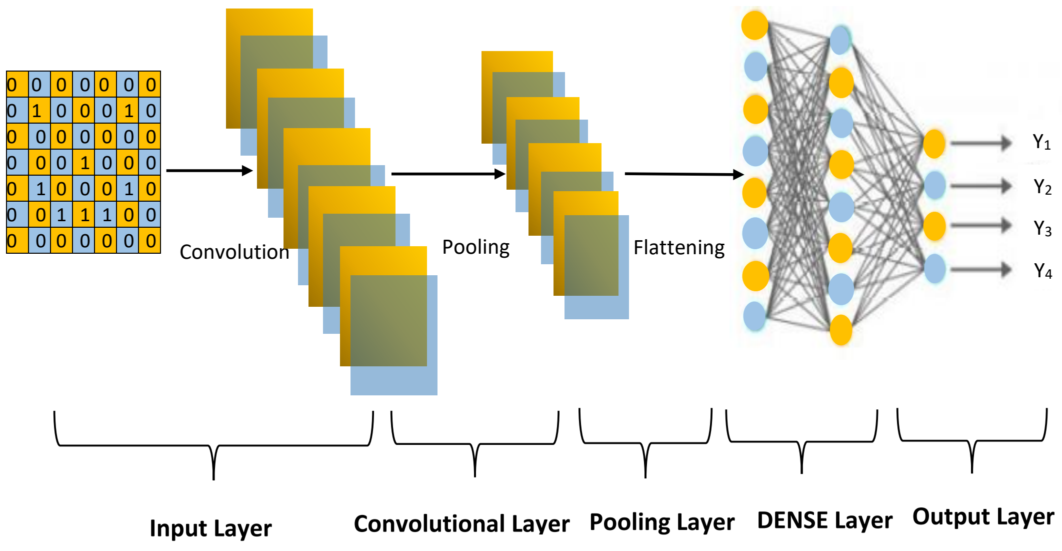

Figure 3. Architecture of convolutional neural network.

The pooling layer commonly comes after the convolutional layer, reduces the size of the feature matrix, expedites training, and minimizes the number of parameters to avoid the overfitting problem. The pooling layer typically follows the convolutional layer and serves to reduce the size of the feature matrix, speed up training, and reduce the number of parameters to prevent overfitting issues [64].

The neurons, weights, and biases in a fully connected (FC) layer play a significant role in establishing connections between the neurons of consecutive layers. These layers are usually the final ones to be added in a CNN architecture and precede the output layer. A significant aspect of CNN models is the activation function, which determines which pieces of information from the model should be forwarded to the output nodes and which should be disregarded, thereby making the network less linear. The activation functions in a CNN model are responsible for determining whether a neuron is activated or not [65].

3.3.2. DENSE

The proposed model incorporates a DENSE layer for enhanced and expedited responses. This layer effectively combines CNN and DBN. The layer has 66 parameters on both the input and output sides. The ReLU (rectified linear activation unit) was utilized in this experiment. To better represent complicated relationships between the inputs and outputs of a neural network, the ReLU function transforms the linear function's output into a nonlinear one. The output of each neuron in a particular layer of a neural network is transmitted into the ReLU function. The output of the ReLU function is then utilized as the input for the subsequent neural network layer. The ReLU function efficiently solves the vanishing gradient problem and is significantly quicker than earlier activation functions. The vanishing gradient problem hinders the network's ability to learn when the gradient of the loss function with respect to the network's weights becomes negligible. Given that the binary derivative of the ReLU function, proves the effectiveness for non-negative values between 0 and 1, it has the potential to address the problem of vanishing gradients in deep neural networks. This property ensures that the gradient does not decrease excessively as the network becomes deeper. It is mathematically represented as:

$$RELU(x) = \begin{cases} 0 & \text{if } x < 0 \\ x & \text{if } x >= 0 \end{cases} \tag{3.2}$$

3.3.3. DBN

The machine learning technology known as a DBN differs significantly from a DNN in several significant ways. They use a feed-forward neural network to process networks and feature a feed-forward architecture with numerous hidden layers. DBNs were designed to replace conventional neural networks during deeply layered network training because of their sluggish learning, the tendency to become caught in local minima due to poor parameter selection, and a necessity for many training datasets [66].

The deep belief network's unsupervised learning algorithm is made up of belief networks and restricted boltzmann machines (RBMs). DBN is a multi-layered belief network, similar to a perceptron and backpropagation neural network, where each layer must be trained before training the DBN as a whole. This is achieved by using the greedy algorithm, which trains the RBMs one by one until they are all trained. DBN has a layered architecture, with the upper two levels being associative memory and the bottom layer consisting of visible units. The relationships between the lower layers are indicated by arrows pointing toward the layer closest to the data.

The lowest layers of the DBN use directed acyclic connections to convert the associative memory into quantifiable data, with the lowest layer of visible units accepting either binary or real data as the input. Like RBM, DBN has no intra-layer connections, with hidden units representing the characteristics that embody the relationships in the data as represented in Figure 4. The two levels are linked by a matrix of proportional weights, W, with the units of each layer being connected to the layer above them [66]. Common activation functions used in DBN include ReLU, SoftMax, tanH, and sigmoid, each with a specific use. Sigmoid and SoftMax functions are typically used for binary classification models, while SoftMax functions are used for multiclass classification using Eq (3.3).

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{3.3}$$

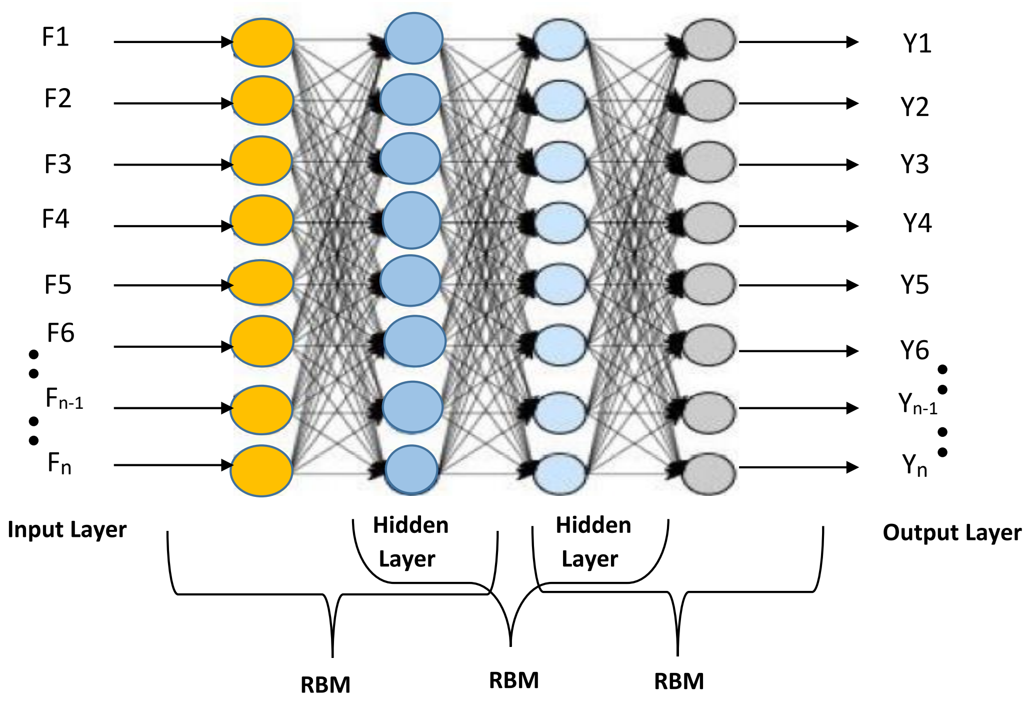

Figure 4. Architecture of deep belief network.

4. Results and discussion

4.1. Experimental setup

The proposed model is implemented using Python, version 3.9, because it is a language that is frequently used for these kinds of studies. We used the Keras package of the TensorFlow library because Keras is easy to implement and it is a structural package. For coding purposes, the Jupyter Notebook was utilized since it displays the results after every code cell. In this experiment, Windows 10 Pro was used with an I5-6th Gen Laptop. The laptop contained a 2.4 GHz processor and 8 GB of RAM.

4.2. Evaluation measures

The model's performance was assessed through an evaluation process. An improvement in the model's results leads to an increase in its performance, while subpar results reveal differences in performance. Researchers often validate the model using accuracy, precision, recall, F1-score, and G-mean score based on the false negative (FN), true negative (TN), false positive (FP), and true positive (TP). The model's performance is evaluated in terms of attack detection using various metrics. During the model development, 80% of the dataset was utilized and pre-processed. The data were then subjected to evaluation and the performance was analyzed based on these metrics [67].

Accuracy in a machine learning classification model refers to the fraction of accurate classifications to the total number of negative and positive observations. Precision measures the accuracy of positive predictions and is equivalent to reliability and positive predictive value. Recall measures the model's capability to identify TP results and is synonymous with the TP rate. The F1-score combines precision and recall to provide a single metric that reflects the model's overall performance. The geometric mean, which is calculated by taking the nth root of the product of all numbers in a set, where n represents the number of observations, provides a single equivalence threshold that combines the true negative rate with the true positive rate.

$$GM = \sqrt[n]{x_1 \times x_2 \times ...x_n} \tag{4.1}$$

4.3. Experimental results

In our study, we carried out experiments for the proposed model using two approaches. The first experiment involved comparing our proposed model with traditional models without utilizing any balancing strategy. In the second experiment, we compared the proposed model with other models using two balancing techniques: SMOTE and SMOTETomek. The proposed model was evaluated using the Multi-Step Cyber-attack dataset.

4.3.1. Without balancing strategy

The performance of the proposed approach was evaluated against the results from [48] in terms of the G-mean score, as depicted in Figure 5. Our proposed approach outperformed the G-mean scores of the Decision Tree (DT) and Random Forest (RF) in [48]. During the training process, five epochs were used. The G-mean score for the proposed approach was 0.813, which is more optimum than [48] scores of 0.710 and 0.650. The model has been enhanced by incorporating the Adam optimizer, with the loss function set as sparse categorical entropy. The activation function of SoftMax was used in the output layer.

The proposed model outperformed all the other traditional models and also outperforms the accuracy score of [47], as shown in Table 3. The comparison between the CNN and the proposed approach is illustrated in Figure 6. The proposed model achieved the highest results for all selected metrics among all the models, as indicated in Table 3. Furthermore, the response time of the proposed model was less compared to other models. The proposed model was tested using 3220 instances with a batch size of 32 and verbose set to 1.

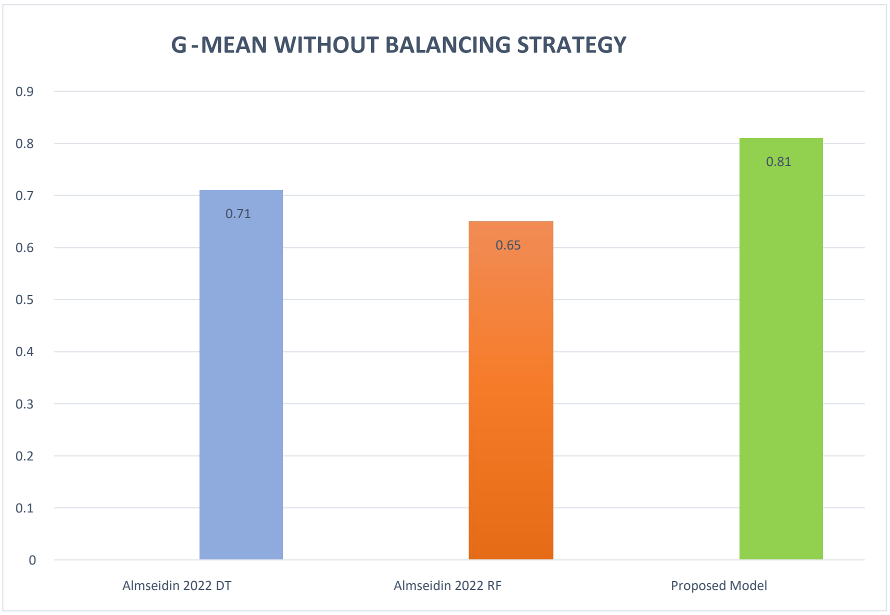

Figure 5. G-mean without balancing strategy.

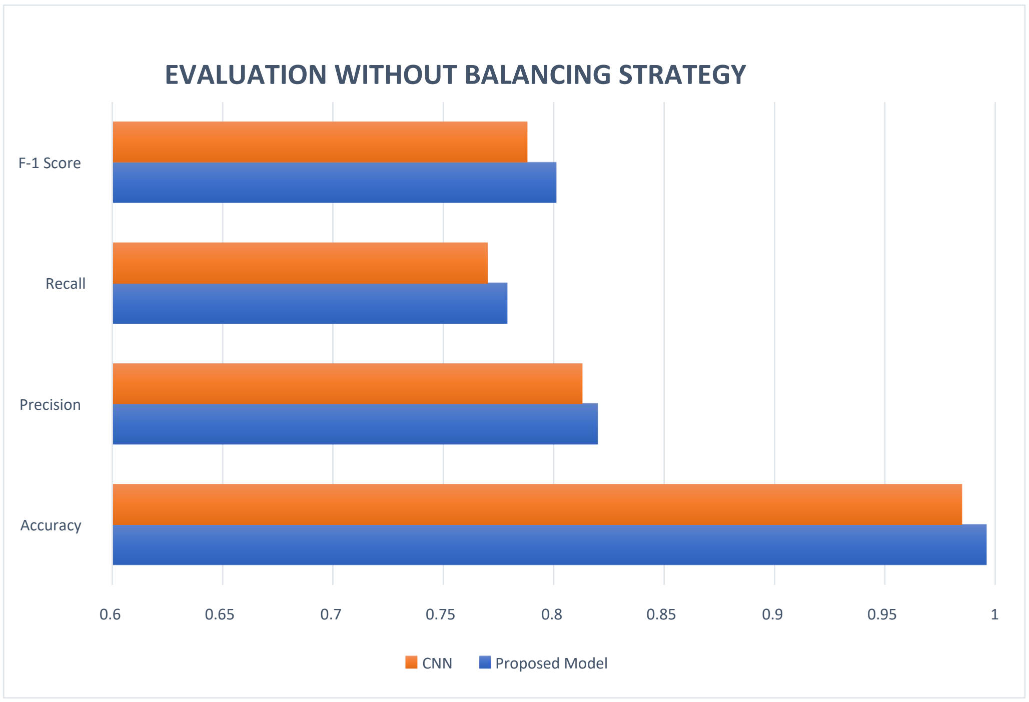

Figure 6. Evaluation without balancing strategy.

Table 3. Scores without balancing strategy.

References	Mode	Accuracy	Precision	Recall	F1-score	G-Mean
[47]	0.825	0.942	0.911	0.926	x	
[48]	DT	x	x	x	x	0.71
	RF	x	x	x	x	0.65
Other techniques	NB	0.787	0.462	0.627	0.313	x
	DNN	0.687	0.114	0.166	0.135	x
	LSTM	0.99	0.708	0.668	0.682	x
	GRU	0.97	0.554	0.522	0.532	x
	CNN	0.993	0.813	0.77	0.788	x
	RNN	0.99	0.777	0.753	0.763	x
Proposed method	CNN-DBN	0.996	0.82	0.779	0.801	0.81

4.3.2. With balancing strategy

The proposed approach was compared with [47] and with [48] using the SMOTE balancing strategy, as shown in Figure 7. The results show that the proposed approach outperforms [47] with higher accuracy, precision, recall, and F1-score and [48] higher G-mean scores. The training process was conducted over five epochs and the G-mean score for the proposed approach was 0.975, while the G-mean scores for [48] using DT and RF were 0.780 and 0.790, respectively. This indicates the superiority of the proposed model. The sparse categorical entropy loss function was utilized in the training process, and the model was tested on the Adam optimizer. The output layer was activated using the SoftMax activation function for multi-class classification.

Table 4. Scores with SMOTE.

References	Model	Accuracy	Precision	Recall	F1-score	G-Mean
[47]		0.825	0.942	0.911	0.926	x
[48]	DT	x	x	x	x	0.78
	RF	x	x	x	x	0.79
Other techniques	NB	0.641	0.644	0.641	0.573	x
	DNN	0.166	0.027	0.166	0.047	x
	LSTM	0.97	0.97	0.971	0.97	x
	GRU	0.943	0.932	0.945	0.943	x
	CNN	0.972	0.971	0.972	0.971	x
	RNN	0.954	0.954	0.956	0.955	x
Proposed method	CNN-DBN	0.976	0.976	0.975	0.975	0.97

The proposed model was compared to other traditional models, as shown in Table 4, and the results indicate that it surpassed their performance. The comparison between the CNN and the suggested model using SMOTE is presented in Figure 8. The proposed model exhibits the highest accuracy, recall, precision, and F1-score score when compared to the other models, as displayed in Table 4. Additionally, it has a faster response time than the other models. The performance of our model was evaluated on the MSCAD dataset, using 13,276 instances with a batch size of 32 and verbose set to 1. The results confirm that the proposed approach outperforms the other models, with 20% of the data being used for testing.

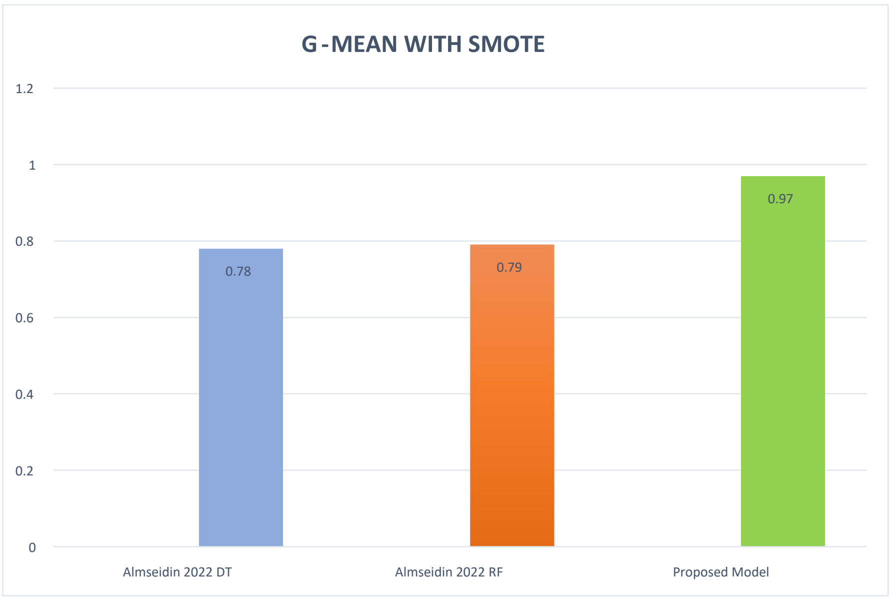

Figure 7. G-Mean with SMOTE.

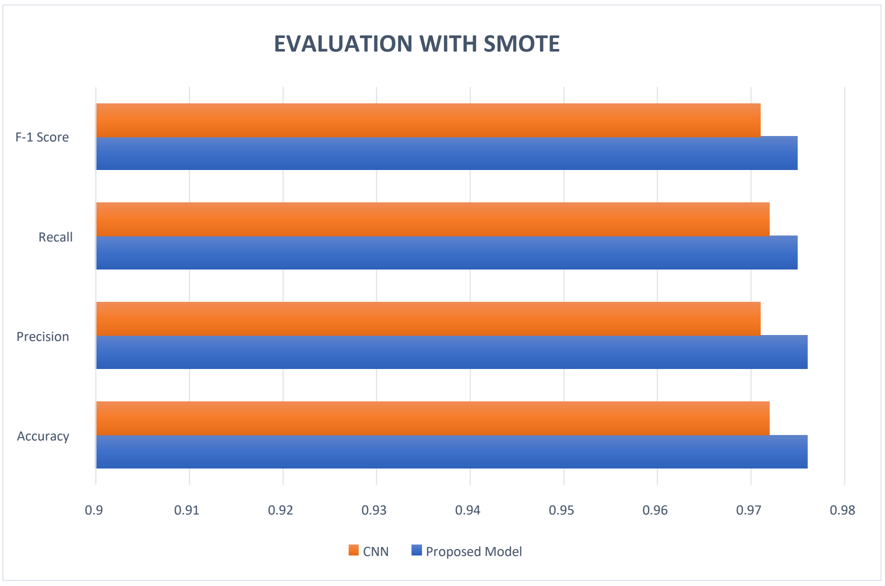

Figure 8. Evaluation with SMOTE.

Our proposed approach surpassed the [47] accuracy, precision, recall, and F1-score and achieved a

SMOTETomek G-mean score of 0.978, as demonstrated in Figure 9. This score succeeds the G-mean scores of both the DT and RF methods described in [48]. The G-mean scores for [48] DT and RF were 0.830 and 0.820, respectively, showcasing the superior performance of our model. The model was trained using the Adam optimizer, and the activation function named SoftMax was utilized in the final layer. The process of training took place over five epochs and utilized the sparse categorical entropy loss function.

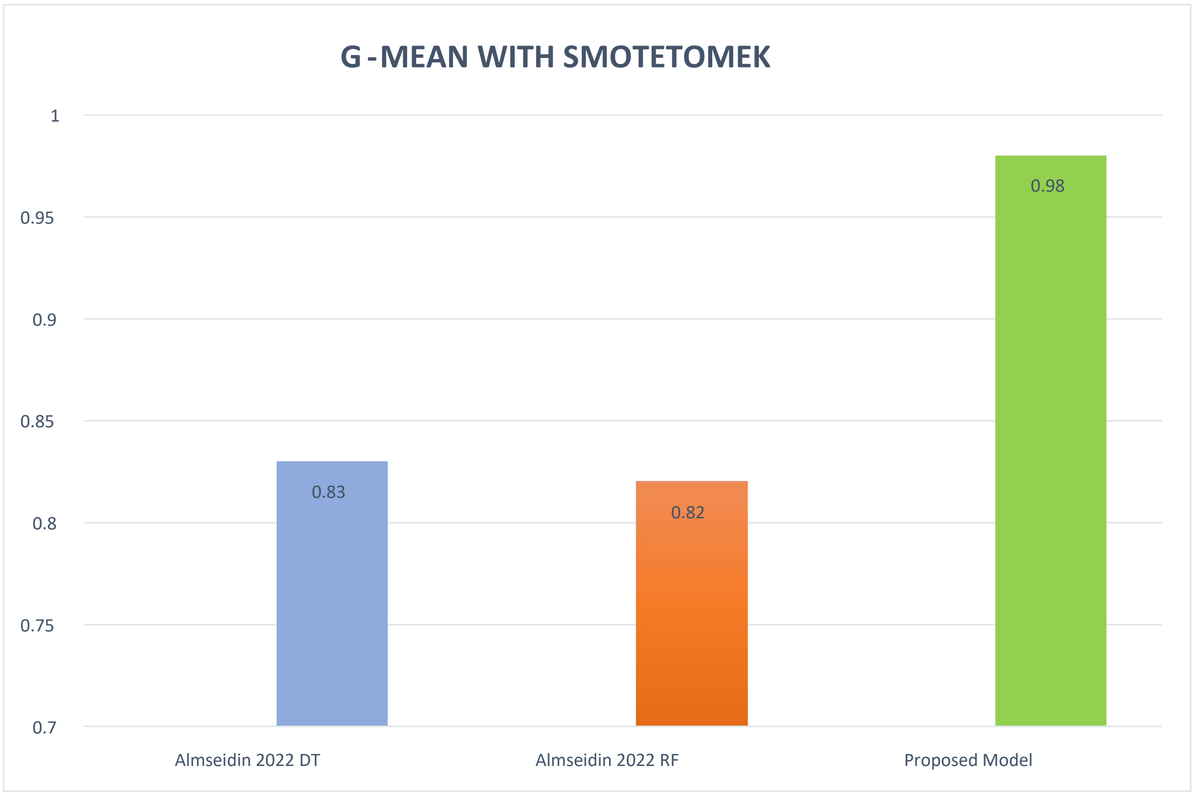

Figure 9. G-Mean with SMOTETomek.

In Figure 10, we compared the performance of the proposed technique with that of a CNN using SMOTETomek. The results are presented in Table 5, which includes the results of both the proposed model and other traditional models. The proposed model outperforms other models in terms of F1-score score, precision, recall, and accuracy, as demonstrated in Table 5. Additionally, our model has a faster response time compared to other models. The model was tested on 13,233 instances using a batch size of 32 and a verbose level of 1 on the MSCAD dataset. The results clearly show that the suggested model produces more optimal scores than other models, even though the scores of the CNN model are slightly lower. However, the CNN model has a longer response time compared to the proposed approach.

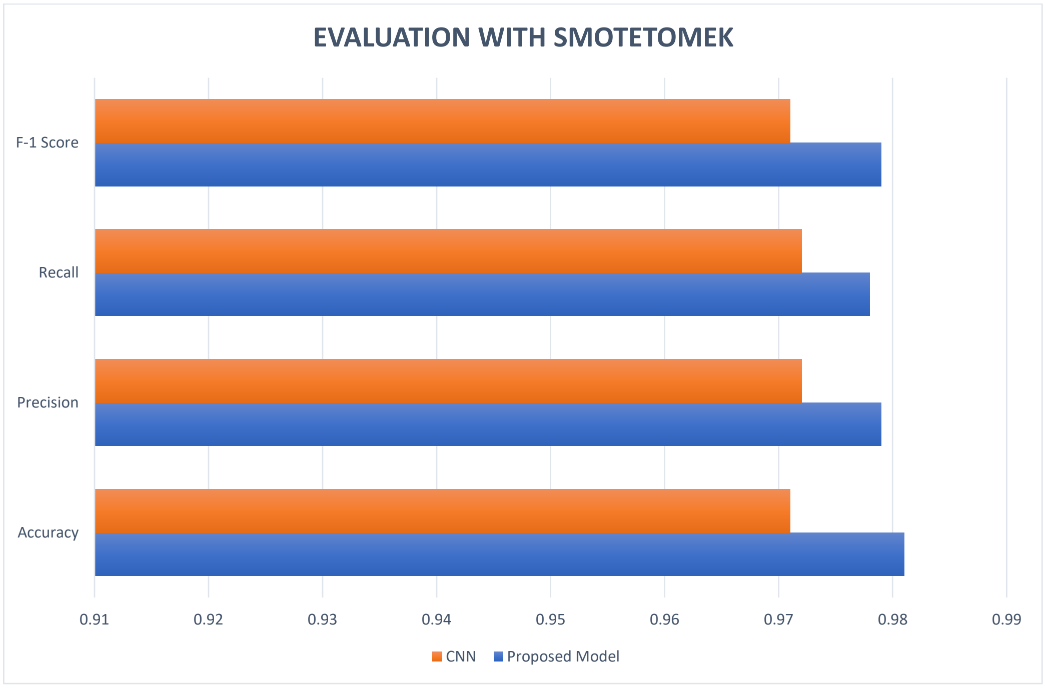

Figure 10. Evaluation with SMOTETomek.

4.4. Discussion

In comparison to previous models, the experimental results demonstrate that the proposed approach provides optimal performance and quicker response time. As shown in Table 3, all scores were evaluated without using any balancing strategy. The CNN accuracy is near to the proposed approach accuracy but it takes more time to execute than our approach. With our proposed methodology, we can also identify binary-class and multi-class attacks. The proposed model has the best accuracy of 99.6%. It also provides a precision of 82%, recall of 77.9%, F1-score of 80.1%, and a G-mean score of 0.813 which provides an improved solution.

The performance of the proposed approach utilizing SMOTE is presented in Table 4. It is evident from the results that the proposed approach outperforms all other models and has a faster execution time. The performance of CNN and LSTM models came closest to the suggested model, but they took much longer to execute. Our model achieved equal precision and accuracy of 97.6%, whereas a recall has a value of 97.5% and a G-mean score of 0.975.

The results of combining SMOTE (oversampling) and Tomek (undersampling), referred to as SMOTETomek, are presented in Table 5. The Table depicts that the proposed model performs more optimally than others in terms of F1-score, G-mean score, recall, accuracy, and precision. Although the CNN model can be compared to the proposed approach, it takes a longer time to execute. The proposed method is suitable for detecting binary as well as multi-class intrusion with a faster response time. In contrast, the proposed model has the best accuracy of 98.1%, provides the best precision of 97.9%, recall of 97.8%, F1-score of 97.9% compared to [47], and the G-mean score of 0.973

compared to [48].

The response time was also calculated for the DL models which were used with the same dataset. Our proposed model responded faster when compared with all the models without using any balancing strategy. The response time for the proposed model is 20 seconds, while for RNN, CNN, GRU, and LSTM, the response time is 41 seconds, 22 seconds, 122 seconds, and 113 seconds, respectively. In contrast to other models, the CNN exhibits a shorter response time. Therefore, we compare the response time of the proposed model with that of the CNN to evaluate its efficiency. By employing SMOTE, the response time for the proposed model is recorded as 14 seconds, while for the CNN, it is 17 seconds. Additionally, when utilizing SMOTETomek, the response time for the proposed model is 13 seconds, whereas, for the CNN, it is 15 seconds. These results clearly demonstrate the effectiveness of the proposed model in terms of response time.

Table 5. Scores with SMOTETomek.

References	Model	Accuracy	Precision	Recall	F1-score	G-Mean
[47]		0.825	0.942	0.911	0.926	x
[48]	DT	x	x	x	x	0.83
	RF	x	x	x	x	0.82
Other techniques	NB	0.643	0.648	0.643	0.574	x
	DNN	0.167	0.278	0.166	0.047	x
	LSTM	0.968	0.97	0.969	0.968	x
	GRU	0.937	0.934	0.939	0.943	x
	CNN	0.971	0.972	0.972	0.971	x
	RNN	0.953	0.955	0.954	0.954	x
Proposed method	CNN-DBN	0.981	0.979	0.978	0.979	0.98

5. Conclusions

The exponential growth of industrial networks and inadequate security assessments pave the way for cyber attackers to cause significant losses and damage to confidential data and sensitive information. A DL-based architecture for intrusion detection in industrial networks is presented in this research. The model employs the use of CNN and DBN algorithms to enhance its performance and reduce both training and response times. The proposed framework's performance was evaluated through experiments on the MSCAD. The experimental findings demonstrated that, in comparison to other conventional models, the suggested approach achieves optimal results with a 99.6% accuracy without using any balancing strategy, 97.6% accuracy using SMOTE, and 98.1% accuracy rate using SMOTETomek for the tested dataset. This extensive research work achieves a high accuracy rate for any anomaly in a short time. It warns the administrator of the industrial network that the system has detected an attack. The administrator then investigates the matter to stop the attacker from getting off the system. The network industry will benefit from this study to design attack-detection devices. In the future, when the IDS detects any intrusion, the same IDS, also prevents the network from intrusion and blocks it. This auto-prevention technique will save the industrial network from losing sensitive information. When the system detects an attack, it should block the next data packet from getting access into the network, thus making the network safe for communication and other industrial purposes. Moreover, other balancing techniques such as SamplePairing, ADASYN, etc will be used to improve the performance of the proposed model in the future as well.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

Acknowledgments

The authors are thankful to the Deanship of Scientific Research at Najran University for funding this work under the Research Groups Funding program grant code (NU/RG/SERC/12/3).

Conflict of interest

The authors declare there is no conflict of interest.

References

1. R. M. Balajee, M. K. J. Kannan, Intrusion detection on AWS cloud through hybrid deep learning algorithm, *Electronics*, **12** (2023), 1423. https://doi.org/10.3390/electronics12061423
2. M. J. Kaur, V. P. Mishra, P. Maheshwari, The convergence of digital twin, IoT, and machine learning: transforming data into action, in *Digital Twin Technologies and Smart Cities*, Springer, (2020), 3–17. https://link.springer.com/chapter/10.1007/978-3-030-18732-3_1
3. O. Abualghanam, H. Alazzam, B. Elshqeirat, M. Qatawneh, M. A. Almaiah, Real-time detection system for data exfiltration over DNS tunneling using machine learning, *Electronics*, **12** (2020), 1467. https://doi.org/10.3390/electronics12061467
4. B. Axelsson, G. Easton, *Industrial Networks (Routledge Revivals): A New View of Reality*, Routledge, 1992.
5. P. C. Smith, L. Hellman, *Small Group Analysis in Industrial Networks*, Routledge, 1992.
6. H. Pourrahmani, A. Yavarinasab, R. Zahedi, A. Gharehghani, M. H. Mohammadi, P. Bastani, et al., The applications of Internet of Things in the automotive industry: a review of the batteries, fuel cells, and engines, *Internet Things*, **19** (2022), 100579. https://doi.org/10.1016/j.iot.2022.100579
7. Y. Yang, K. McLaughlin, T. Littler, S. Sezer, H. F. Wang, Rule-based intrusion detection system for SCADA networks, in *2nd IET Renewable Power Generation Conference*, 2013. https://doi.org/10.1049/cp.2013.1729
8. M. Baezner, P. Robin, *Stuxnet*, Report, Center for Security Studies (CSS), ETH Zürich, 2017. Available from: https://www.research-collection.ethz.ch/handle/20.500.11850/184547.
9. Zagaris, Bruce, Boggess, Kenneth, Cybercrime, HeinOnline, 2021. Available from: https://heinonline.org/HOL/LandingPage?handle=hein.journals/ielr37&div=152.
10. E. D. Knapp, J. T. Langill, *Industrial Network Security: Securing Critical Infrastructure Networks for Smart Grid, SCADA, and Other Industrial Control Systems*, Elsevier, 2015.

11. S. Hong, C. Lv, T. Zhao, B. Wang, J. Wang, J. Zhu, Cascading failure analysis and restoration strategy in an interdependent network, *J. Phys. A: Math. Theor.*, **49** (2016), 195101. https://doi.org/10.1088/1751-8113/49/19/195101

12. A. Kwasinski, W. Weaver, P. L. Chapman, P. T. Krein, Telecommunications power plant damage assessment for hurricane Katrina–site survey and follow-up results, *IEEE Syst. J.*, **3** (2009), 277–287. https://doi.org/10.1109/JSYST.2009.2026783

13. R. M. Lee, M. J. Assante, T. Conway, Analysis of the cyber attack on the Ukrainian power grid, *Electr. Inf. Sharing Anal. Cent.*, **388** (2016), 1–29.

14. J. Angséus, R. Ekbom, *Network-Based Intrusion Detection Systems for Industrial Control Systems*, Master's thesis, University of Gothenburg, Gothenburg, 2017.

15. H. Y. Kwon, T. Kim, M. K. Lee, Advanced intrusion detection combining signature-based and behavior-based detection methods, *Electronics*, **11** (2022), 867. https://doi.org/10.3390/electronics11060867

16. Y. Jia, M. Wang, Y. Wang, Network intrusion detection algorithm based on deep neural network, *IET Inf. Secur.*, **13** (2019), 48–53. https://doi.org/10.1049/iet-ifs.2018.5258

17. F. Rustam, M. F. Mushtaq, A. Hamza, M. S. Farooq, A. D. Jurcut, I. Ashraf, Denial of service attack classification using machine learning with multi-features, *Electronice*, **11** (2022), 3817. https://doi.org/10.3390/electronics11223817

18. N. Naz, M. A. Khan, S. A. Alsuhibany, M. Diyan, Z. Tan, M. Almas Khan, et al., Ensemble learning-based IDS for sensors telemetry data in IoT networks, *Math. Biosci. Eng.*, **19** (2022), 10550–10580. https://doi.org/10.3934/mbe.2022493

19. S. Agrawal, S. Sarkar, O. Aouedi, G. Yenduri, K. Piamrat, S. Bhattacharya, et al., Federated learning for intrusion detection system: Concepts, challenges and future directions, *arXiv preprint*, (2022), arXiv:2106.09527. https://doi.org/10.48550/arXiv.2106.09527

20. M. Almseidin, M. Alkasassbeh, An accurate detection approach for IoT botnet attack using interpolation reasoning method, *Information*, **13** (2022), 300. https://doi.org/10.3390/info13060300

21. F. Zhai, T. Yang, H. Chen, B. He, S. Li, Intrusion detection method based on CNN–GRU–FL in a smart grid environment, *Electronics*, **12** (2023), 1164. https://doi.org/10.3390/electronics12051164

22. M. Cheminod, L. Durante, A. Valenzano, Review of security issues in industrial networks, *IEEE Trans. Ind. Inf.*, **9** (2013), 277–293. https://doi.org/10.1109/TII.2012.2198666

23. S. Hong, J. Zhu, L. A. Braunstein, T. Zhao, Q. You, Cascading failure and recovery of spatially interdependent networks, *J. Stat. Mech: Theory Exp.*, **2017** (2017). https://doi.org/10.1088/1742-5468/aa8c36

24. I. Butun, M. Almgren, V. Gulisano, M. Papatriantafilou, Intrusion detection in industrial networks via data streaming, in *Industrial IoT*, Springer, (2020), 213–238. https://doi.org/10.1007/978-3-030-42500-5_6

25. L. Zang, D. Ma, A hybrid approach toward efficient and accurate intrusion detection for in-vehicle networks, *IEEE Access*, **10** (2022), 10852–10866. https://doi.org/10.1109/ACCESS.2022.3145007
26. 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** (2019), 41525–41550. https://doi.org/10.1109/ACCESS.2019.2895334
27. G. M. D. Teyou, J. Ziazet, Convolutional neural network for intrusion detection system in cyber-physical systems, *arXiv preprint*, (2019), arXiv:1905.03168. https://doi.org/10.48550/arXiv.1905.03168
28. X. Wang, S. Yin, H. Li, J. Wang, L. Teng, A network intrusion detection method based on deep multi-scale convolutional neural network, *Int. J. Wireless Inf. Networks*, **27** (2020), 503–517. https://doi.org/10.1007/s10776-020-00495-3
29. S. Ullah, J. Ahmad, M. A. Khan, E. H. Alkhammash, M. Hadjouni, Y. Y. Ghadi, et al., A new intrusion detection system for the Internet of Things via deep convolutional neural network and feature engineering, *Sensors*, **22** (2022), 3607. https://doi.org/10.3390/s22103607
30. S. Hong, T. Yue, H. Liu, Vehicle energy system active defense: a health assessment of lithium-ion batteries, *Int. J. Intell. Syst.*, **37** (2022), 10081–10099. https://doi.org/10.1002/int.22309
31. M. Cheminod, L. Durante, A. Valenzano, Review of security issues in industrial networks, *IEEE Trans. Ind. Inf.*, **9** (2012), 277–293. https://doi.org/10.1109/TII.2012.2198666
32. S. D. D. Anton, S. Sinha, H. D. Schotten, Anomaly-based intrusion detection in industrial data with SVM and random forests, *arXiv preprint*, (2019), arXiv:1907.10374. https://doi.org/10.48550/arXiv.1907.10374
33. Z. Wang, Z. Li, D. He, S. Chan, A lightweight approach for network intrusion detection in industrial cyber-physical systems based on knowledge distillation and deep metric learning, *Expert Syst. Appl.*, **206**, (2022), 117671. https://doi.org/10.1016/j.eswa.2022.117671
34. S. Potluri, S. Ahmed, C. Diedrich, Securing industrial control systems from false data injection attacks with convolutional neural networks, in *Development and Analysis of Deep Learning Architectures*, Springer, (2020), 197–222. https://doi.org/10.1007/978-3-030-31764-5_8
35. S. Potluri, S. Ahmed, C. Diedrich, Convolutional neural networks for multi-class intrusion detection system, in *Mining Intelligence and Knowledge Exploration*, Springer, (2018), 225–238. https://doi.org/10.1007/978-3-030-05918-7_20
36. Y. Zhu, Y. Zi, J. Xu, Transfer learning-based SAE-CNN for industrial data processing in multiple working conditions recognition, in *2022 IEEE International Conference on Prognostics and Health Management (ICPHM)*, (2022), 167–172. https://doi.org/10.1109/ICPHM53196.2022.9815720
37. T. Cruz, L. Rosa, J. Proença, L. Maglaras, M. Aubigny, L. Lev, et al., A cybersecurity detection framework for supervisory control and data acquisition systems, *IEEE Trans. Ind. Inf.*, **12** (2016), 2236–2246. https://doi.org/10.1109/TII.2016.2599841

38. S. Huda, J. Yearwood, M. M. Hassan, A. Almogren, Securing the operations in SCADA-IoT platform based industrial control system using ensemble of deep belief networks, *Appl. Soft Comput.*, **71** (2018), 66–77. https://doi.org/ 10.1016/j.asoc.2018.06.017

39. J. Jiao, X. J. Zheng, Fault diagnosis method for industrial robots based on DBN joint information fusion technology, *Comput. Intell. Neurosci.*, **2022** (2022). https://doi.org/10.1155/2022/4340817

40. K. Lu, G. Zeng, X. Luo, J. Weng, W. Luo, Y. Wu, Evolutionary deep belief network for cyber-attack detection in industrial automation and control system, *IEEE Trans. Ind. Inf.*, **17** (2021), 7618–7627. https://doi.org/10.1109/TII.2021.3053304

41. A. A. Suzen, Developing a multi-level intrusion detection system using hybrid-DBN, *J. Ambient Intell. Hum. Comput.*, **12** (2021), 1913–1923. https://doi.org/10.1007/s12652-020-02271-w

42. S. Zhang, J. Lai, Q. Yao, Traffic anomaly detection model of electric power industrial control based on DBN-LSTM, in *2021 IEEE 23rd Int Conf on High Performance Computing, Communications; 7th Int Conf on Data Science, Systems; 19th Int Conf on Smart City; 7th Int Conf on Dependability in Sensor, Cloud, Big Data Systems, Application*, (2021), 1902–1907. https://doi.org/10.1109/HPCC-DSS-SmartCity-DependSys53884.2021.00284

43. G. Meena, R. R. Choudhary, A review paper on IDS classification using KDD 99 and NSL KDD dataset in WEKA, in *2017 International Conference on Computer, Communications and Electronics (Comptelix)*, (2017), 553–558. https://doi.org/10.1109/COMPTELIX.2017.8004032

44. L. Whaley, The critical institutional analysis and development (CIAD) framework, *Int. J. Commons*, **12** (2018). https://doi.org/10.18352/ijc.848

45. P. Foremski, C. Callegari, M. Pagano, Waterfall: Rapid identification of IP flows using cascade classification, in *Computer Networks*, (2014), 14–23. https://doi.org/10.1007/978-3-319-07941-7_2

46. R. Zuech, T. Khoshgoftaar, N. Seliya, M. M. Najafabadi, C. Kemp, A new intrusion detection benchmarking system, in *Proceedings of the Twenty-Eighth International Florida Artificial Intelligence Research Society Conference*, 2015.

47. K. M. A. Alheeti, A. Alzahrani, O. H. Jasim, D. Al-Dosary, H. M. Ahmed, M. S. Al-Ani, Intelligent detection system for multi-step cyber-attack based on machine learning, in *2023 15th International Conference on Developments in eSystems Engineering (DeSE)*, (2023), 510–514. https://doi.org/10.1109/DeSE58274.2023.10100226

48. M. Almseidin, J. Al-Sawwa, M. Alkasassbeh, Generating a benchmark cyber multi-step attacks dataset for intrusion detection, *J. Intell. Fuzzy Syst.*, **43** (2022), 3679–3694. https://doi.org/10.3233/JIFS-213247

49. S. Suthaharan, T. Panchagnula, Relevance feature selection with data cleaning for intrusion detection system, in *2012 Proceedings of IEEE Southeastcon*, (2012), 1–6. https://doi.org/10.1109/SECon.2012.6196965

50. M. Bahrololum, E. Salahi, M. Khaleghi, Machine learning techniques for feature reduction in intrusion detection systems: A comparison, in *2009 Fourth International Conference on Computer Sciences and Convergence Information Technology*, (2009), 1091–1095. https://doi.org/10.1109/ICCIT.2009.89

51. J. W. Osborne, *Best Practices in Data Cleaning: A Complete Guide to Everything You Need to Do Before and After Collecting Your Data*, SAGE Publications, 2013. https://doi.org/10.4135/9781452269948

52. W. McKinney, Pandas: A foundational Python library for data analysis and statistics, *Python High Perform. Sci. Comput.*, **14** (2011), 1–9.

53. K. Farhana, M. Rahman, M. T. Ahmed, An intrusion detection system for packet and flow-based networks using a deep neural network approach, *Int. J. Electr. Comput. Eng.*, **10** (2020), 5514–5525. https://doi.org/10.11591/ijece.v10i5.pp5514-5525

54. D. T. Dantas, H. Li, T. Charton, L. Chen, R. Zhang, Machine learning based anomaly-based intrusion detection system in a full digital substation, in *15th International Conference on Developments in Power System Protection*, 2020. https://doi.org/10.1049/cp.2020.0049

55. W. Wang, X. Zhang, S. Gombault, S. J. Knapskog, Attribute normalization in network intrusion detection, in *2009 10th International Symposium on Pervasive Systems, Algorithms, and Networks*, (2009), 448–453. https://doi.org/10.1109/I-SPAN.2009.49

56. A. Tesfahun, D. L. Bhaskari, Intrusion detection using random forests classifier with SMOTE and feature reduction, in *2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies*, (2013), 127–132. https://doi.org/10.1109/CUBE.2013.31

57. B. Yan, G. Han, M. Sun, S. Ye, A novel region adaptive SMOTE algorithm for intrusion detection on imbalanced problem, in *2017 3rd IEEE International Conference on Computer and Communications (ICCC)*, (2017), 1281–1286. https://doi.org/10.1109/CompComm.2017.8322749

58. J. Han, W. Pak, High performance network intrusion detection system using two-stage LSTM and incremental created hybrid features, *Electronics*, **12** (2023), 956. https://doi.org/10.3390/electronics12040956

59. J. Kim, J. Kim, H. Kim, M. Shim, E. Choi, CNN-based network intrusion detection against denial-of-service attacks, *Electronics*, **9** (2020), 916. https://doi.org/10.3390/electronics9060916

60. M. Azizjon, A. Jumabek, W. Kim, 1D CNN-based network intrusion detection with normalization on imbalanced data, in *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, (2020), 218–224. https://doi.org/10.1109/ICAIIC48513.2020.9064976

61. S. Albawi, T. A. Mohammed, S. Al-Zawi, Understanding of a convolutional neural network, in *2017 International Conference on Engineering and Technology (ICET)*, (2017), 1–6. https://doi.org/10.1109/ICEngTechnol.2017.8308186

62. Q. Zhang, M. Zhang, T. Chen, Z. Sun, Y. Ma, B. Yu, Recent advances in convolutional neural network acceleration, *arXiv preprint*, (2019), arXiv:1807.08596. https://doi.org/10.48550/arXiv.1807.08596

63. R. Vinayakumar, K. P. Soman, P. Poornachandran, Applying convolutional neural network for network intrusion detection, in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, (2017), 1222–1228. https://doi.org/10.1109/ICACCI.2017.8126009

64. P. Liu, An intrusion detection system based on convolutional neural network, in *Proceedings of the 2019 11th International Conference on Computer and Automation Engineering*, (2019), 62–67. https://doi.org/10.1145/3313991.3314009

65. N. Gupta, P. Bedi, V. Jindal, Effect of activation functions on the performance of deep learning algorithms for network intrusion detection systems, in *Proceedings of ICETIT 2019*, Springer, (2020), 949–960. https://doi.org/10.1007/978-3-030-30577-2_84

66. H. Jia, J. Liu, M. Zhang, X. He, W. Sun, Network intrusion detection based on IE-DBN model, *Comput. Commun.*, **178** (2021), 131–140. https://doi.org/10.1016/j.comcom.2021.07.016

67. S. Ullah, M. A. Khan, J. Ahmad, S. S. Jamal, Z. Huma, M. T. Hassan, et al., HDL-IDS: a hybrid deep learning architecture for intrusion detection in the Internet of Vehicles, *Sensors*, **22** (2022), 1340. https://doi.org/10.3390/s22041340