Neural Networks 165 (2023) 321–332

Contents lists available at ScienceDirect

Neural Networks

journal homepage: www.elsevier.com/locate/neunet

SCADA securing system using deep learning to prevent cyber infiltration

Sayawu Yakubu Diaba [a,*], Theophilus Anafo [b], Lord Anertei Tetteh [c], Michael Alewo Oyibo [d], Andrew Adewale Alola [e,f], Miadreza Shafie-khah [g], Mohammed Elmusrati [a]

[a] Department of Telecommunication Engineering, School of Technology and Innovations, University of Vaasa, Vaasa, Finland
[b] Department of Electrical/Electronic Engineering, Cape Coast Technical University 5P3+F7H, Cape Coast, Ghana
[c] Department of Electrical and Electronic Engineering, Koforidua Technical University, 3P8P+F5F, Koforidua, Ghana
[d] National Bureau of Statistics, Abuja, Nigeria
[e] CREDS-Centre for Research on Digitalization and Sustainability, Inland Norway University of Applied Sciences, Norway
[f] Faculty of Economics, Administrative and Social Sciences, Nisantasi University, Istanbul, Turkey
[g] Department of Electrical Engineering, School of Technology and Innovations, University of Vaasa, Vaasa, Finland

ARTICLE INFO

Article history:
Received 15 February 2023
Received in revised form 14 May 2023
Accepted 23 May 2023
Available online 2 June 2023

Keywords:
Genetically seeded flora
Intrusion detection systems
Long short-term memory
Recurrent neural network
Residual neural network
And transformer neural network

ABSTRACT

Supervisory Control and Data Acquisition (SCADA) systems are computer-based control architectures specifically engineered for the operation of industrial machinery via hardware and software models. These systems are used to project, monitor, and automate the state of the operational network through the utilization of ethernet links, which enable two-way communications. However, as a result of their constant connectivity to the internet and the lack of security frameworks within their internal architecture, they are susceptible to cyber-attacks. In light of this, we have proposed an intrusion detection algorithm, intending to alleviate this security bottleneck. The proposed algorithm, the Genetically Seeded Flora (GSF) feature optimization algorithm, is integrated with Transformer Neural Network (TNN) and functions by detecting changes in operational patterns that may be indicative of an intruder's involvement. The proposed Genetically Seeded Flora Transformer Neural Network (GSFTNN) algorithm stands in stark contrast to the signature-based method employed by traditional intrusion detection systems. To evaluate the performance of the proposed algorithm, extensive experiments are conducted using the WUSTL-IIOT-2018 ICS SCADA cyber security dataset. The results of these experiments indicate that the proposed algorithm outperforms traditional algorithms such as Residual Neural Networks (ResNet), Recurrent Neural Networks (RNN), and Long Short-Term Memory (LSTM) in terms of accuracy and efficiency.

1. Introduction

Industry 4.0, also known as the Fourth Industrial Revolution (Smith & Fressoli, 2021), began in the early 2000s (Montalban, Iradier, & Member, 2020) as a result of advancements in internet communication and the development of automated software and frameworks (Hoffmann Souza, da Costa, de Oliveira Ramos, & da Rosa Righi, 2021). This has made it possible to control the manufacturing process using simple computer programs and microcontrollers, leading to increased customization of products (Chen & Chang, 2020; Rousopoulou et al., 2022). Research is ongoing to develop self-decision-making control systems for manufacturing processes and to enable remote monitoring and control of these processes (Hassan Malik, Alam, Kusik, & Moullec, 2020; Jasperneite, Sauter, & Wollschlaeger, 2020; Sarker, 2022). SCADA systems are mainly used to control and monitor (Kumar & S, 2020), components of vital infrastructures (Kirubakaran, 2020), such as smart grids, pipelines, transportation, telecommunication, and manufacturing plants (Lee & Hong, 2020). The SCADA systems can also act as a status projector for monitoring the operation of the Industrial Control Systems (ICS). It can be integrated with a Programmable Logic Controller (PLC) and other control technologies like Proportional Integral Derivative (PID) controllers (V, 2020). SCADA devices have a high operational speed, enabling real-time data analysis.

The high integration of communication infrastructure in the smart grid and the connections to the internet (Cherifi & Hamami, 2018; Yang, McLaughlin, Sezer, Yuan, & Huang, 2014) in the

* Corresponding author.
E-mail address: sdiaba@uwasa.fi (S.Y. Diaba).
URL: http://dx.doi.org/10.1016/j.cviu.2017.00.000 (S.Y. Diaba).

https://doi.org/10.1016/j.neunet.2023.05.047

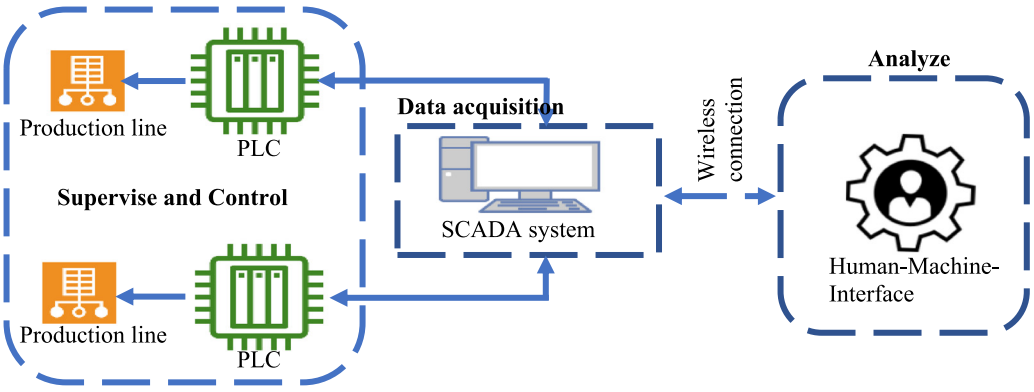

Fig. 1. The architectural overview of a SCADA system.

Table 1
Comparison between SCADA and IoT.

Parameters	SCADA	IoT
Communication medium	Semi-wireless	Wireless
Storage	Local	Cloud
System integration	Limited integration to the peripheral	Easily integrate with the peripheral
Operational reliability	High	Low
Suitability	Suitable for a big production line	Suitable for minor applications
Threat possibility	Medium to high	High

SCADA system have created a lacuna for cyber-attacks (Altunay, Albayrak, Ozalp, & Cakmak, 2021; Singh, Ebrahem, & Govindarasu, 2019). A technology designed with the sole purpose of granting cyber protection for computers, computer networks, data transmissions, and legitimate access is termed *Cybersecurity*. The cyber-security systems are primarily a setup of computer (host) security systems and network security systems. Each of these has, at minimum, antivirus software, a firewall, and Intrusion Detection System (IDS) (Singh, Garg, Kumar, & Saquib, 2015).

SCADA system security concerns are receiving more attention as the frequency of security incidents against these crucial infrastructures is rising (Samdarshi, Sinha, & Tripathi, 2016). Though, the presence of cyber threats in SCADA systems are comparatively less compared to the Internet of Things (IoT) systems because the SCADA networks are not connected to an open internet as the IoT systems (see Fig. 1).

Yet, the SCADA systems are in the third top position in terms of attaining frequent threat disturbances amongst the other applications. The cyber-attacks are targeted at the SCADA systems for manipulating the operational control of the network. The hackers may cause damage to the power system when they have control over the switches, isolators, and relays under the command of the SCADA systems (see Table 1).

These attacks' repercussions may jeopardize the safety, availability, reputation, profitability, and reliability of the targeted organizations (Liu & Wang, 2022). Traditional SCADA systems were not created with a cyber-securing protocol, albeit at the moment some models do have certain firewall security measures activated. The SCADA systems are still a target for hackers (Altaha, Lee, Aslam, & Hong, 2020) who exploit the firewall's weaknesses. To protect the communication infrastructure of the smart grid, it is essential to develop a SCADA network intrusion detection solution that considers both operational requirements and particular traffic characteristics of SCADA systems (see Table 2).

Motivated by the above facts, the following are regarded as the encapsulation of the primary contributions of this paper:

- We investigate the application of deep learning techniques in detecting cyber threats in both industrial and general settings with a specific focus on SCADA systems used in the smart grid. We explore the potential of these techniques in identifying and mitigating cyber-attacks in various environments, highlighting their effectiveness and limitations. We also investigate the possible integration of deep learning techniques with existing security systems to enhance their performance and overall security.
- We propose an algorithm for detecting cyber intrusions by analyzing changes in operational patterns that are related to intrusion activity. To achieve this, we utilize a GSFTNN algorithm that is custom-made for the specific task of intrusion detection. The algorithm is designed to identify anomalies in the operational patterns and flag them as potential intrusions.
- We perform extensive simulations using the WUSTL-IIOT-2018 ICS SCADA cyber security dataset to evaluate the effectiveness of deep learning techniques in detecting cyber-attacks in SCADA systems. The simulation process consists of two stages: binary classification and multiclass classification. In the binary classification stage, the data is classified as normal and attacks. In the multiclass detection stage, the data is further classified into three categories: exploiting attacks, aggressive attacks, and normal traffic. The results of the simulation are used to analyze the effectiveness of the deep learning techniques and to pinpoint possible areas for development.

The remainder of the paper is structured as follows; related studies are presented in Section 2. The methodology is in Section 3 where we give the data description as well as the types of attacks and the summary of the attacks therein. In Section 4, experimental analysis is presented and finally, the paper's conclusion is presented in Section 5.

2. Related studies

In the area of SCADA security, older papers have investigated the use of machine learning techniques to enhance security. For example, the authors of Maglaras and Jiang (2014) proposed a machine learning-based approach for detecting anomalies in SCADA systems and compares the performance of several different algorithms, including neural networks, support vector machines, and decision trees. The authors proposed a novel method for detecting intrusions in the SCADA system, which can identify abnormal activity even if an attacker attempts to conceal it in the control layer of the system. To assess the effectiveness of the algorithms, supervised machine learning models were examined to categorize normal and abnormal behaviors in an ICS. The authors

Table 2
Some types of attacks involved in the SCADA system.

Attack types	Reflection	Initiation
Denial of Service	Enforcing maximum traffic to the network to block the actual communication	Poor authentication platform
Ransomware attacks	Malfunction and operational block of PLCs	Vulnerable hardware
Malicious node attacks	Execution of unauthorized operation	Web interface with an outdated operating system
Phishing attacks	Control over the SCADA system	Absence of network isolation and weak authentication
Worm attacks	Blocks access/operation	No network isolation
Honeypot attacks	Reframe the device function	Weak servers and vulnerable policies on security

used several machine learning models in examining the models, and they performed well at spotting abnormalities, particularly stealthy attacks. According to the findings, random forest outperforms other classifier algorithms (Mokhtari, Abbaspour, Yen, & Sargolzaei, 2021).

In Lopez Perez, Adamsky, Soua, and Engel (2018) a machine learning approach for intrusion detection in SCADA systems was accessed on a real-world dataset. The authors find that the random forest detects intrusion effectively. Older papers demonstrate that machine learning has been a topic of interest in the area of SCADA security for at least a decade and that the use of machine learning for enhancing SCADA security is not a new idea. The authors of Teixeira et al. (2018) looked at cyber-attacks that use AI-based techniques and found some mitigation techniques that can be used to stop such attacks. Also, they examined current trends in AI-based cyber-attacks and were able to identify the methodologies and strategies currently used in executing AI-based cyber-attacks as well as what future scenarios will likely be conceivable to control such attacks.

Several studies have investigated the use of artificial neural networks (ANN), convolutional neural networks (CNN), and RNN to detect and prevent cyber-attacks in SCADA systems. These methods have been demonstrated to be successful in detecting and preventing a wide range of cyber-attacks, including malware, phishing, and distributed denial-of-service attacks (Al Husaini, Habaebi, Hameed, Islam, & Gunawan, 2020; Balla, Habaebi, Islam, & Mubarak, 2022; Khan, Zhang, Alazab, & Kumar, 2019). A CNN (P, Hong, Gao, Yao, & Zhang, 2020; Wu, Hong, & Chanussot, 2022) defining the significant temporal patterns of SCADA communication and pinpointing time windows that are vulnerable to network attacks rather than hand-crafted characteristics for specific network packets or flows was proposed. The authors provided a re-training method to manage instances of network attacks that have never been detected before. The study utilized actual SCADA traffic datasets and the results demonstrate that the deep-learning-based technique that has been proposed is suitable for SCADA systems' network intrusion detection, attaining high detection accuracy and offering the capacity to address newly emerging threats (Altaha et al., 2020; Yang, Cheng, & Chuah, 2019).

The use of deep learning for enhancing the security of SCADA systems has been a growing area of research in recent years. Studies that focus on deep learning, suggest that this area of research has advanced significantly (Gao et al., 2023; Wu, Hong, & Chanussot, 2023), and that deep learning is a promising direction (Yang & Chen, 2019) for enhancing the security of SCADA systems. With the increasing use of technology in critical infrastructures, such as medical devices, power plants, and water treatment facilities (Lee & Hong, 2020; Pliatsios, Sarigiannidis, Lagkas, & Sarigiannidis, 2020), the need for robust and secure SCADA systems is more pressing than ever. Cyber-attacks on SCADA systems can result in significant harm, including disruption of essential services, loss of sensitive information, and physical damage to equipment. To address these concerns, many researchers have turned to deep learning as a promising solution for enhancing the security of SCADA systems (Avola, Cinque, Fagioli, & Foresti, 2022).

The research in Wang, Harrou, Bouyeddou, Senouci, and Sun (2022) presented a stacked deep learning-driven method for detecting cyber-attacks. The relevant aspects of the suspicious behaviors were thoroughly learned by the proposed stacked deep learning model, which then distinguishes them from normal actions. As a result, the stacked deep learning-based intrusion detection approach performs better than some cutting-edge shallow methods, such as the standalone deep learning models, naive Bayes, random forests, nearest neighbor, oneR, AdaBoost, and support vector machine. The research in Jmila and Houda (2022) focuses more on shallow classifiers, which are still often employed in machine learning-based IDS because of their maturity and ease of usage. The authors tested the resistance to various adversarial approaches often utilized in the state-of-the-art of AdaBoost, bagging, decision tree, gradient boosting, logistic regression, random forest, support vector classifier, and even a deep learning network. A Gaussian data augmentation defensive method was implemented and its impact on increasing classifier robustness was assessed. The findings demonstrate that not all classifiers are affected equally by attacks, that a classifier's robustness relies on the attack, and that depending on the network intrusion detection scenario, a trade-off between performance and robustness must be considered.

It is worth noting that while deep learning has shown great promise in enhancing SCADA security, there are still many challenges to overcome. For example, deep learning models can be vulnerable to adversarial attacks (Jmila & Houda, 2022; Ozdag, 2018) and the quality of training data can significantly impact the performance of these models. Nevertheless, the research in this area suggests that deep learning is a promising direction for enhancing the security of SCADA systems. It has the potential to enhance the security of SCADA systems in a variety of ways, including detecting and preventing cyber-attacks, mitigating system failures, protecting sensitive information, and enhancing the security of communication networks. However, as with any new technology, there are still many challenges to overcome, such as improving the robustness of deep learning models, addressing the issue of data scarcity, and developing secure deep learning systems that are resistant to adversarial attacks.

3. Methodology

Addressing cyber intrusion in SCADA is the main motive of the proposed algorithm and it is implemented in this paper with a deep learning-based approach. The algorithm is a hybrid of GSF and TNN and it is compared to ResNet, RNN, and LSTM models for identifying the best-performing algorithm in detecting intrusions in SCADA systems. Washington University St. Louis-Industrial IoT-2018 (WUSTL-IIOT-2018) dataset for ICS SCADA cybersecurity used in Ahakonye, Nwakanma, Lee, and Kim (2023) is the dataset used in this study to examine the efficiency and accuracy of the above-mentioned algorithms.

3.1. Data

The WUSTL-IIOT-2018 ICS SCADA is a collection of network traffic data captured from a real-world ICS that was intentionally

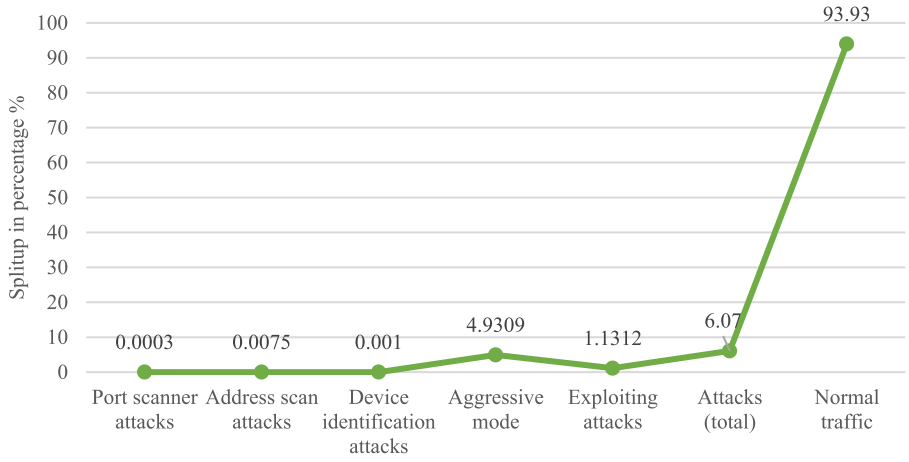

Fig. 2. The summary of attacks available in the dataset.

subjected to cyber-attacks. The dataset was created to evaluate intrusion IDS in cyber–physical systems. It was collected from a water treatment testbed in the United States, a representative example of a real-world industrial control system. To generate the intrusion data, the testbed is connected to a network monitoring system accompanied by a scan tool, and its features are summarized to form a dataset. The following are the attacks available in the dataset along with their data generation procedure.

3.1.1. Port scanner attack

The port scanner attacks are included in the SCADA system for observing its active ports in the operation and control process. To generate such attacks, certain targeted nodes are generated at different frequencies of time using the Nmap tool. At the same time, the Transmission Control Protocol (TCP) connection is also partially disabled for allowing the attack to generate the data attributes in the testbed.

3.1.2. Address scanner attack

The address scan attacks are generated to observe the Modbus server address. This allows the attacker to reach the connected hardware devices of the SCADA system for implementing the malfunctioning algorithm. In general, the SCADA systems are engaged with only one Modbus address and it gives an open path to the intruder for generating different types of attacks when it is tracked.

3.1.3. Device identification attack

The attackers are creating the device identification attack to find out the specification and model numbers of the connected devices to the SCADA network. It can be produced by tracking the Modbus slave identification of the targeted SCADA system. Thus, the vulnerable hardware devices must be verified regularly in the SCADA systems. In some cases, device identification attacks are avoided by having an additional authentication process.

3.1.4. Aggressive model device attack

In the aggressive mode of device identification attack, the information of all the slave buses is collected along with the Modbus slave identification. These kinds of attacks are employed in the SCADA network to freeze all the connected hardware without sending any malicious nodes. In real-time applications, each connected hardware is implemented with a separate authentication process. This improves the complexity of the security algorithm and restricts the success rate of aggressive mode attacks.

Table 3
Sample of the employed dataset.

Sport	Tpkt	Tbyte	Spkt	Dpkt	Sbyte	Tgt
143	2	180	2	0	180	0
68	2	684	2	0	684	0
0	1	60	1	0	60	0
61 845	20	127	10	10	644	0
61 846	20	127	10	10	644	0
44 287	6	372	4	2	248	1
48 456	20	128	12	8	776	1
48 458	20	139	12	8	782	1
44 460	20	128	12	8	776	1
61 850	12	780	6	6	396	0
61 849	12	780	6	6	396	0
61 848	18	1152	10	8	644	0

3.1.5. Exploit attack

In exploit attacks, information about the operational state of PLC coils is obtained to understand how the SCADA system is currently functioning. This allows attackers to replicate the manufacturing process and produce identical products. A system for generating and using inspection records was used to observe the movements of normal and malicious nodes in a testbed model during the dataset generation process. The testbed model was made to run continuously for 25 h to monitor the changes in the network. Fig. 2 presents the summary of attacks available in the dataset utilized.

Table 3 includes several features that describe various aspects of network communications. One of these features is the source port (sport), which represents the number of unique source ports. Another feature is the total packets (TotPkt), which represent the total number of packets involved in the communications. Additionally, the total bytes (TotBytes) feature indicates the total number of bytes transferred. Two other features included in Table 3 are the source packets (SrcPkts) and destination packets (DstPkts). These features represent the number of packets transmitted from the source and the number of packets received at the destination, respectively. Finally, the Source bytes (SrcBytes) feature indicates the number of bytes transferred from the source to the destination during communications. Together, these features provide a comprehensive picture of the different aspects of network communications that can be used to analyze and understand network traffic patterns (see Fig. 3).

3.2. Preprocessing

The total number of traffic data available in the dataset is 7 037 983 counts. In that 427 206 instances are attack-oriented

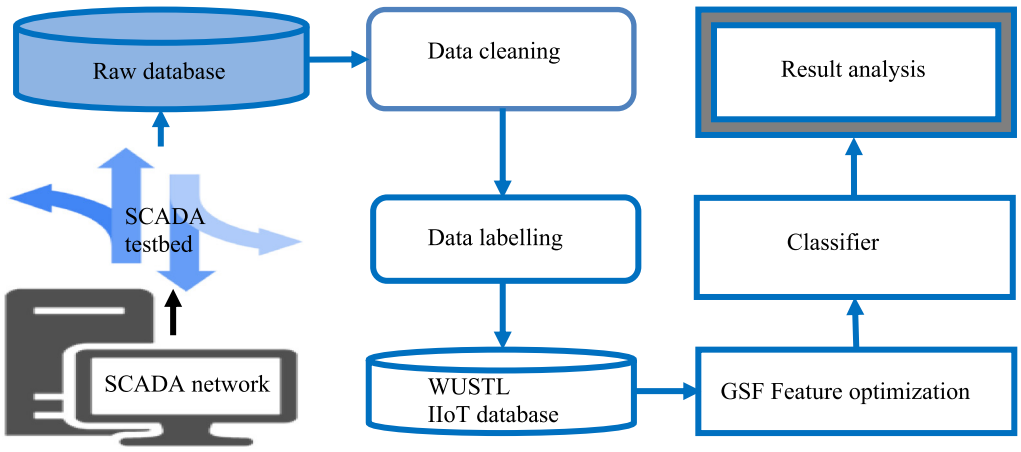

Fig. 3. The design display of the proposed model.

Table 4
Summary of the dataset.

Data type	Timetable
Number of variables	6
Number of rows	7 037 983
Number of variables with missing	0
Number of variables with duplicate	6
Timestamp is regularly spaced	True
Timestamp has missing	False
Timestamp has duplicates	False
Timestamp sorted	True

and the remaining 6 610 777 belong to the normal traffic category. In the proposed algorithm, a dataset split ratio of 70:30 is employed after cross-validation of five folds is implemented in the phase 1 simulations, the proposed algorithm is utilized to identify the normal and attack traffics, and in phase 2, the proposed algorithm was experimented to categorize the traffics as normal traffic, exploiting attacks and aggressive mode attack.

The workflow followed in the proposed algorithm does not have any pre-processing step as the data available in the dataset are already pre-processed using the data cleaning and labeling process. However, the raw data values collected from the network monitoring tools may have some missing and error data due to sudden fluctuations in its operation. Therefore, a data cleaning (data cleaner app in MATLAB 2022b) process was used to manipulate the missing values in the dataset. The data cleaner app in MATLAB 2022b is an interactive tool for locating messed-up column-oriented data, cleaning numerous variables of data at once, and improving the cleaning process. A total of 7 037 983 samples and 6 variables dataset was loaded into the data cleaner app. We set the data cleaner app to use only standard indicators to detect missing values such as not a number (NaN), not a time (NaT), and cell of character vectors. The remove missing method is used to remove the data rows with missing entries. The outliers are another type of error usually present in the data with an unusual entry. To handle it, we used the fill outlier cleaning method with linear interpolation as the filling method. The method of detection is the moving mean and the threshold was set at 3. Tables 4 and 5 explore the data summary after processing.

3.2.1. Feature optimization

The feature optimization process is employed in this work to handle the abnormalities in the available dataset. In some cases, the feature attributes may remain almost the same on different attack data. Hence it worsens the misclassification and reduces the precision level of the classifier system. A GSF feature optimization algorithm is utilized in the proposed algorithm to avoid such limitations. GSF is an upgraded version of an artificial flora optimization technique that selects the connecting point relevancy based on the seed-growing property of the respective points. The GSF model is equipped with a genetic algorithm for estimating the best seed-growing points. The genetic algorithm estimates the location by analyzing the propagation distance among the points along with the plant weights. The propagation distance d_y of the seeds is predicted using the equation written in (Cheng, Wu, & Wang, 2018; Selvarajan, Shaik, Ameerjohn, & Kannan, 2020).

$$d_y = d_{y1}\,(\psi \times j_1) + d_{y2}\,(\psi \times j_2) \tag{1}$$

where j_1 and j_2 stand for searching coefficients. The uniform random numbers between 0 and 1 are generated by *rand* and denoted by ψ. The grandparent's propagation distance and the parent's propagation distance are denoted by d_{y1} and d_{y2} respectively. The two main steps of the flora optimization technique are the spreading and selection behavior. Thus, the position of the plant is determined using the matrix $P_{i,y}$, the dimension is denoted by i and y represents the total number of plants in the flora. The equation for the spreading process can be written as

$$P_{i,y} = \psi \times d\,(2 - d) \tag{2}$$

where d is for the maximum limit area. Since the weight value may be determined by the standard deviation of the propagation distance between the parent plant and offspring plant when updating the plant position, we can express the equation as

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}\left(P_{i,y} - P'_{i,y}\right)^2}{N}} \tag{3}$$

The present position of the offspring plant is estimated as

$$P'_{i,y^*j} = D_{i,y^*j} + P_{i,y} \tag{4}$$

where, the position of the original plant is denoted by $P_{i,y}$, j is the maximum number of seeds that a single plant can produce, D_{i,y^*j} is for the random estimation of Gaussian distribution value with zero mean and j variance. The final best estimation of the offspring plant is estimated by the probability of survival given as

$$P = \left|\sqrt{\frac{F\left(P'_{i,y^*j}\right)}{F_{\max}}}\right| * Q_x^{y^*j-1} \tag{5}$$

Table 5
Data summary after preprocessing.

	Missing count	Minimum	Maximum	Mean	Median	Mode	Standard deviation
Time	0	00:00:00	69:46:38	34:53:19	53.09	00:00:00	20 000
Sport	0	0	42 238	0.00003589	35 458	354	0.00621
TotPkt	0	1	96 057	9.4518	2	2	66 196
TotBytes	0	60	98 180 744	0.0003317	124	124	4.156e−5
SrcPkts	0	1	96 057	8.0302	2	2	
DstPkts	0	0	4881	1.6534	0	0	35.7538
SrcBytes	0	60	6 942 046	915.1435	124	124	4.81e−4

The selective probability is represented by Q_x and the value of Q_x must fall within the range of 0 and 1. According to the authors of Cheng et al. (2018), having a higher value of Q_x is desirable for problems that can easily get into a local optimal solution. $P'_{i,y*j}$ represents the fitness of the jth solution. F_{max} represents the flora's maximum fitness. The best characteristics among the overall attributes are chosen from this phase and sent to the classification step. In doing so, the classifier automates the rule-generation process to predict the class label. This improves classification accuracy.

3.3. Classifier

Classifiers are like algorithmic filters used to segregate the given data samples into their respective classes based on the instructions learned from their training samples. The learning process of a classifier model can be refined by implementing a customized preprocessing or feature selection model. The neural network algorithm assigns the testing sample into a particular class based on their similarity score calculated from the comparison. The proposed algorithm explores the following classifiers on the SCADA cyber-attack dataset to find the most suitable model for the real-time application. Classifiers are used to classify data samples into different classes based on the instructions learned from the training samples. Many different types of classifiers can be used for various applications. Some common types include:

Decision Trees: These classifiers use a tree-like structure to make decisions. Each internal node represents a feature of the input data, and each leaf node represents a class label. The algorithm starts at the root node and follows the branches based on the feature values of the input data until a leaf node is reached, which determines the class label of the input.

Naive Bayes: This is a probabilistic classifier that makes class predictions based on the probability of each class given the input features. The "naïve" part of the name refers to the assumption that the features are independent of each other, which is not always true in real-world data.

Neural Networks: These classifiers use a network of artificial neurons to make class predictions. The neurons are organized into layers, with the input layer receiving the input features, one or more hidden layers processing the information, and the output layer producing the class predictions. The network learns to make accurate predictions by adjusting the weights of the connections between the neurons.

Random Forest: This ensemble technique combines different decision trees to make class predictions. Each tree is trained using a different random subset of the input data, and the consensus of all the trees is used to get the final prediction.

Support Vector Machine: This is a type of linear classifier that finds the best boundary (or hyperplane) to separate the input data into different classes. The algorithm is based on finding the line that maximizes the margin, which is the distance between the boundary and the closest points from each class.

In the case of SCADA cyber-attack dataset, it is crucial to find the most suitable model that can quickly and accurately identify cyber-attacks in real-time. The proposed algorithm may compare the performance of these different classifiers on the dataset and select the one that achieves the highest accuracy or lowest false-positive rate.

3.4. Transformer neural network

The TNN algorithm was proposed in the year 2017 to overcome the limitation of computational complexity in many neural network algorithms. It is achieved by utilizing the Graphic Processing Unit (GPU) sources effectively by processing the input data simultaneously. Therefore, the time required for the training process is also limited in the TNN. TNNs are structured with a multi-headed attention layer for learning the input data that allows processing the data in a parallel process. However, in the traditional RNN, the values are considered in sequential order. The TNN is very efficient in natural language processing problems and data mining problems, the architecture is shown in Fig. 4. The encoder and decoder are the two major blocks involved in the TNN architecture and it has a positional encoding block right in front of the encoder block. The role of the positional embedding block is to determine the value of the inputs denoted by (x) at different attributes' places.

In the proposed algorithm the features are counted from '$F_1 \ldots F_5$'. The value of x at F_1may not have the same weight at F_3. The values on certain features may remain the same even if the output classes are different. The positional encoding model addresses such issues and assigns the weights of x into a unique parameter while storing it in the neurons of the TNN. The encoder consists of multi-head attention and a feed-forward block, where the multi-head attention (Selvarajan et al., 2020) has a pair of sub-layers. The input parameters are learned by the multi-head attention block in terms of queries, keys, and range format. The collected parameters are operated with a learnable linear transformation for n times. A constant value is applied in this block as a tuning parameter for operating it to the product of the query with all keys. The output values from the blocks are observed from a SoftMax function from the value of its corresponding weight. The attention outputs are linearly gathered to form a final output where a normalization step is added. Hence the residual connection of the input data is estimated.

$$Q_{ini} = xW^{Q}_{inp} \tag{6}$$

$$K_{inp} = xW^{K}_{inp} \tag{7}$$

$$V_{inp} = xW^{V}_{inp} \tag{8}$$

where, x represents the input, and W represents the customized constant value of the input. The query, keys, and value parameters of the input are represented by Q, K, and V respectively.

$$head_{inp} = \alpha\left(Q_{inp} * K_{inp} * V_{inp}\right) \tag{9}$$

$$\alpha\left(Q_{inp} * K_{inp} * V_{inp}\right) = softmax\left(\frac{Q_{inp} * K^{T}_{inp}}{\sqrt{d}}\right) V_{inp} \tag{10}$$

Algorithm of the GSF feature model:

input = data features (F_1 to F_5) extracted from the testbed
output = selected index of the attributes, F_{sel}
flora seed initialization, F_{si} = rand [size (F_1 to F_5)]
seed-in location of the plants, $P_{i,j} = F_{si}$ (d) (2-d)
propagation distance is taken from d_y
X, Y are representing the feature sizes
Initialize j = 1
  for i = 1 to X*Y
    offspring plants are created by $P_{i,y*j}$
    calculate fitness function
probability estimation on the best plant is observed from 'P'
    when P < rand, then
      F_{sel} (i) = P
reproduction and mutation
j = j + 1
    else
      $P_{i,j} = P_{i,j-1}$
    end if population replacement
end 'i' loop

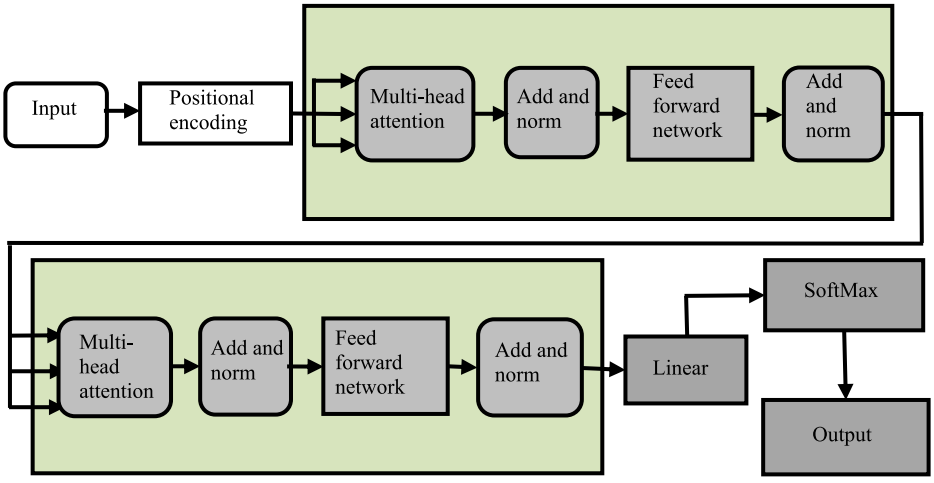

Fig. 4. The architecture of the TNN.

$$norm = x + head_{inp} \tag{11}$$

$$head^2_{inp} = norm\ \left(x + head_{inp}\right) \tag{12}$$

The output attributes obtained from the multi-head attention blocks are moved to a feed-forward network (FFN) for observing an improved output. The normalization process is also included in the output of FFNs and the value of FFN is analyzed as follows

$$FFN = \gamma\ (\delta)\, W_1 + b_1 \tag{13}$$

$$x_{out} = norm(head^2_{inp} + FFN) \tag{14}$$

where γ represents the rectified linear product and δ is for the gated recurrent block.

3.5. Recurrent neural network

The RNN models were developed to regularize the data movement inside the neural network. In the traditional neural network models, the input parameters were allowed to move from one neuron to another neuron without considering anything. As a result, some neurons are unaware of the status of other attributes taken from the input. The RNN regularizes this by making all the attributes follow a sequence movement inside the neural networks. The involvement of the hidden layer makes the neurons store the hidden information regarding the previous attributes, so a small amount of data storage is allocated to each neuron. In some cases, the RNN models are implemented with more than one hidden layer block. There the weight and bias of each hidden will get change from each other to store the different feature information from the given input. Hence the layers included between the input and output layers are independent and do not consider the formation of other hidden layers. The independence of hidden layer weights and biases is making the RNN more complex than their previous models. In some applications, the weights and biases are regularized with the same value, improving computational efficiency. The current state of the neurons is analyzed by

$$Cur_s = f\ (Cur_{s-1}, Inp_s) \tag{15}$$

where Cur_s represents the present state and Inp_s denotes the input state. The activation function of the current state is applied as Eq. (15) and the output is predicted by Eq. (16).

$$Cur_s = \tanh\left(W_{rec} Cur_{s-1} + W_{inp} Inp_s\right) \tag{16}$$

$$Out_s = \tanh\left(W_{out} Cur_s\right) \tag{17}$$

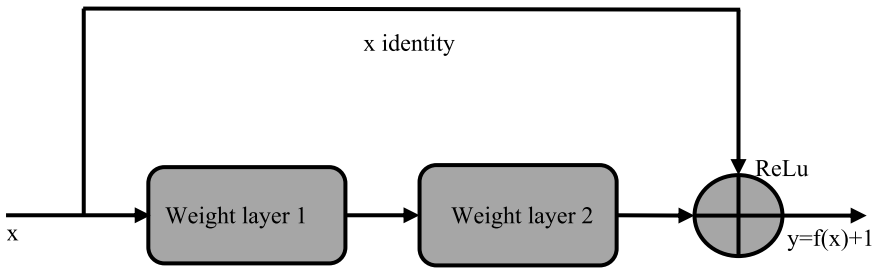

Fig. 5. The architecture of a residual learning.

3.6. Long short-term memory

The LSTM network (Karim, Fazle, Majumdar, & Darabi, 2019) was developed to address the vanishing and exploding gradients problems found in RNN. The LSTM networks were trained to erase the irrelevant information stored in the neuron from the given input. It is achieved by implementing the network with a customized activation function called gates. The internal cell state of the neurons is having the useful information extracted from the training data which is required for the upcoming operation. The LSTM network reads the state of the input gate, modulation gate, forget gate, and output gate by calculating it with element-wise multiple vectors of the given input. Then the neurons will erase the information of the forget gate and combine the information of the input and modulation gate to form output as

$$Cur_s = \left(G_{inp} * G_{inp}^{mod}\right) + \left(G_{for} * Cur_{s-1}\right) \quad (18)$$

where G_{inp} takes gate input, G_{inp}^{mod} stands for gate-modulated input and it is G_{for} for forget gate

3.7. Residual network

ResNet models were proposed to observe more complex features from the given input attributes with a greater number of hidden layers. Each layer in the ResNet is allowed to take some specific feature from the given input. The main idea behind ResNet is to allow the network to learn residual functions, rather than learning the full mapping from input to output. In a residual network, each layer has a shortcut connection that bypasses one or more layers and directly connects the input of the current layer to the output. This allows the network to learn the residual, or the difference between the input and the output of the layer, which is easier to optimize than the full mapping. The residual functions are then added to the output of the corresponding layer, allowing the network to effectively learn the full mapping. ResNet has shown impressive results on a wide range of computer vision tasks, including image classification, object detection, and semantic segmentation. Its architecture has inspired many subsequent neural network models and has become a benchmark for deep learning research (He, Zhang, Ren, & Sun, 2016a, 2016b). However, in some cases, the ResNet was giving poor accuracy by having some unwanted features in its operations. It is addressed by the recent year ResNets by adding dropout and regularization blocks. The architectural view of the ResNet is shown in Fig. 5 with its operational outcome.

4. Experimental analysis and discussion

The experimental work is performed in two phases, binary classification, and multiclass classification. In binary classification, the given information is segregated as normal and attacks. In multiclass detection, the data are classified as exploiting attacks, aggressive attacks, and normal traffic. The performances of the

Table 6
The hyperparameter setting.

Parameter	Total
Validation scheme	Cross-validation
Cross-validation folds	5 folds
Epoch	1000
Activation functions	Softmax, ReLu
Maximum number of split	100
Split criterion	Gini's diversity index

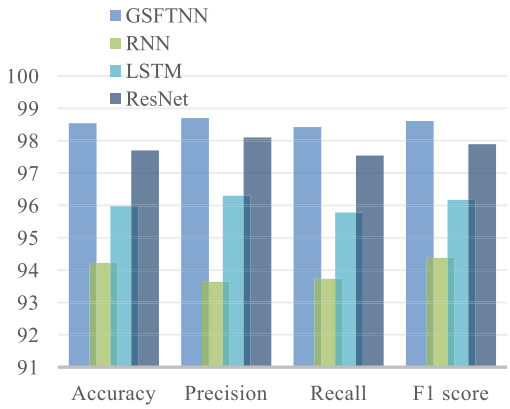

Fig. 6. Phase 1 performance analysis on the verified algorithms.

GSFTNN, RNN, LSTM, and ResNet models are verified with their accuracy, precision, recall, detection rate, and f1 score in both phases (see Tables 6 and 7).

Tables 8 and 9 are representing the performances of the verified algorithms in phase 1 and phase 2 respectively.

The results presented in Fig. 6 demonstrate that the proposed GSFTNN model achieves a high accuracy of 98.54%, indicating its ability to correctly classify a significant majority of the cases. In comparison, the RNN model achieves an accuracy of 94.22%, while the LSTM and ResNet models exhibit accuracy rates of 95.98% and 97.7%, respectively. The proposed GSFTNN model also shows an average precision of 98.7% across all normal and attack categories, outperforming the RNN (93.64%), LSTM (96.3%), and ResNet (98.1%) models. The recall values were 98.42% (Proposed GSFTNN), 93.73% (RNN), 95.78% (LSTM), and 97.54% (ResNet). The F1 score provides a well-rounded evaluation of system performance. In the case of the proposed GSFTNN model, the F1 score is reported as 98.61%, with the RNN, LSTM, and ResNet achieving scores of 94.38%, 96.17%, and 97.89%, respectively.

The result depicted in Fig. 7 indicates the proposed GSFTNN model attained an accuracy of 99.12%, outperforming the comparative models, which attained accuracies of 96.4% (RNN), 97.25% (LSTM), and 98.1% (ResNet), respectively. Additionally, the proposed model attained an average precision of 99.26%, while the precision values for RNN, LSTM, and ResNet were 96.82%, 97.57%, and 98.33%, respectively. The recall values were 98.85% (Proposed

Table 7
Data split up into phases.

Phase 1 – (2 Class)			Phase 2 – (3 Class)		
Classes	Training	Testing	Classes	Training	Testing
Normal traffic	3 966 466	2 644 311	Normal traffic	3 966 466	2 644 311
Attacks	256 324	170 882	Exploiting attacks	47 769	31 845
			Aggressive mode attacks	208 222	138 814

Table 8
Performance of the verified algorithms on the phase 1 dataset.

Algorithms	Accuracy	Precision	Recall	F1 score
GSFTNN	98.54	98.7	98.42	98.61
RNN	94.22	93.64	93.73	94.38
LSTM	95.98	96.3	95.78	96.17
ResNet	97.7	98.1	97.54	97.89

Table 9
Performance of the verified algorithms on the phase 2 dataset.

Algorithms	Accuracy	Precision	Recall	F1 score
GSFTNN	99.12	99.26	98.85	99.2
RNN	96.4	96.82	93.73	96.5
LSTM	97.25	97.57	96.97	97.41
ResNet	98.1	98.33	97.74	98.26

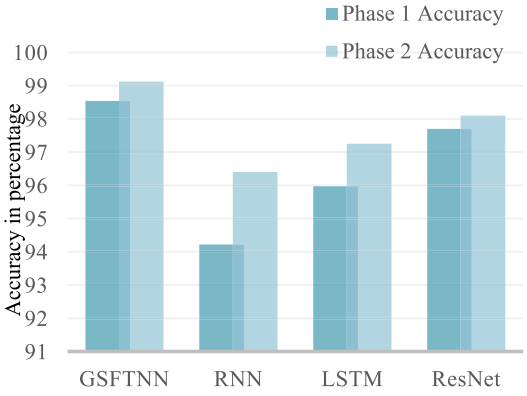

Fig. 8. Comparative analysis of accuracies at both phases.

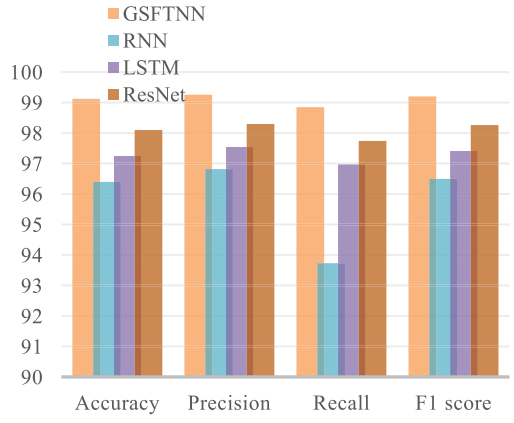

Fig. 7. Phase 2 performance analysis on the verified algorithms.

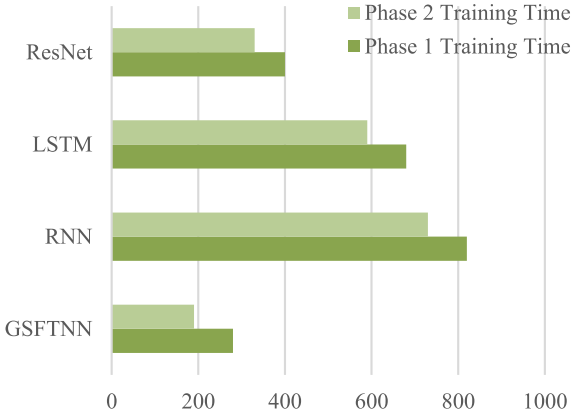

Fig. 9. Comparative analysis of training time with both phases.

GSFTNN), 93.73% (RNN), 96.97% (LSTM), and 97.74% (ResNet). Finally, the GSFTNN model achieved an F1 score of 99.2%, while the RNN, LSTM, and ResNet models scored 96.5%, 97.41%, and 98.26%, respectively.

Fig. 8 indicates the accuracy comparison between the phase 1 and the phase 2 analyses. The phase 2 accuracies are comparatively high in all the algorithms because the attack classes in phase 2 contain only 2 attacks but in phase 1 the data attack model count is 5. The two major attack classes are considered in phase 2, whereas in phase 1 the minor classes with fewer sample counts were considered for the analysis. It indicates that all the classifiers are performing well when their sample counts are high for the training process. The phase 1 accuracy can also be improved if the remaining 3 data sample counts are averaged using some data augmentation process.

The performance of the proposed GSFTNN model shows better accuracy on both phase operations. This is achieved because of its multi-head attention block. At the same time, the performance of its previous model RNN shows a lesser accuracy rate when compared to all the other models due to its sequential operation process. Also, the performances of the LSTM show a slighter improvement in its experiment by eradicating unwanted information from its neurons. The ResNet models are very efficient in general but their nature of having more layer count makes the model suffer from getting the optimum features for the analytic process. The training time attainments of the verified algorithms are shown in Fig. 9 where the performances of GSFTNN indicate a betterment due to the nature of the simultaneous operation. All the algorithms are showing a betterment in the phase 2 model where the sample counts are comparatively minimum than the phase 1 operation.

Zooming on to measure the effectiveness of the proposed algorithm, further comparative analysis of the four (4) deep learning algorithms was conducted. Figs. 10 and 11 illustrate the confusion matrix and the receiver operating characteristic (ROC), respectively.

A confusion matrix is a table that is used to evaluate the performance of a classifier by comparing the predicted classes with the true classes. It is a useful tool for understanding the strengths and weaknesses of a classifier, and it can be used to identify areas for improvement. The matrix is made up of four quadrants that represent the number of true positives, false positives, true negatives, and false negatives. These values can

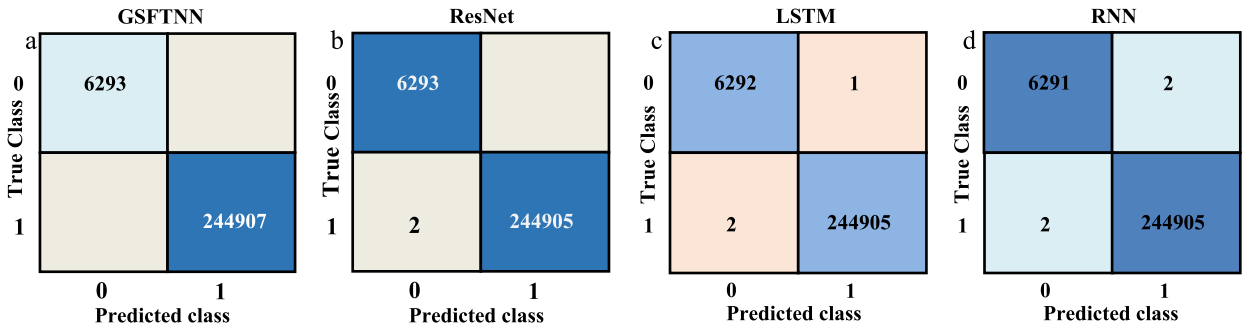

Fig. 10. (a) Confusion matrix of the GSFTNN; (b) Confusion matrix of the ResNet; (c) Confusion matrix of the LSTM; (d) Confusion matrix of the RNN.

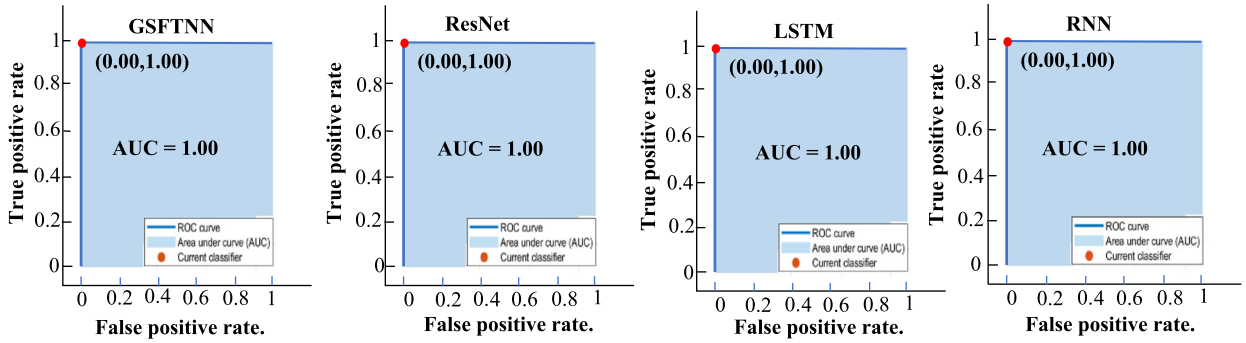

Fig. 11. (a) ROC of the GSFTNN; (b) ROC of the ResNet; (c) ROC of the LSTM; (d) ROC of the RNN.

Table 10
Comparison of existing algorithms.

Reference	Year	Dataset	Algorithm	Accuracy	Recall	F1 score	Precision
Altaha et al. (2020)	2020	Generated dataset	CNN	98.1			
		Generated dataset	FNN	98.8			
		Generated dataset	GRU	98.1			
		Generated dataset	LSTM	98.0			
		Generated dataset	RNN	98.0			
Chen, Dewi, Huang, and Caraka (2020)	2020	Bank marketing dataset	RF+SVM	89.0	91.37		97.91
		Bank marketing dataset	RF+KNN	88.6	90.8		96.91
		Bank marketing dataset	RF+RF	90.99	91.22		98.10
Khoei, Aissou, Hu, and Kaabouch (2021)	2021	CICDDoS-2019	KNN	94.6	94.4		
		CICDDoS-2019	RF	94.0	94.0		
		CICDDoS-2019	Stacking	96.0	97.3		
		CICDDoS-2019	Naïve Bayes	87.0	77.1		
Abdelkhalek and Govindarasu (2022)	2022	WUSTL-IIoT-2018	ANN	98.40	98.02	98.97	**99.57**
	2023	WUSTL-IIoT-2018	**GSFTNN**	**99.12**	**98.85**	**99.2**	99.26

then be used to calculate various performance metrics such as f1 score, recall, precision, and accuracy. A binary classifier system's performance as the discrimination threshold changes are graphically depicted by a ROC curve. The genuine positive rate (sensitivity) against the false positive rate (specificity) at various threshold settings is plotted on the ROC curve. By comparing a classifier algorithm's performance to that of a random guessing classifier, it is frequently possible to gauge how well it performs. The area under curve (AUC) is a frequently used performance statistic for classifiers. While a classifier that performs no better than random guessing has an AUC of 0.5, a perfect classifier has 1. ROC curves are frequently used to compare the effectiveness of various classifiers or the effectiveness of a single classifier in various scenarios (see Table 10).

5. Conclusion

Cybersecurity in the smart grid has become critically important on a multi-stakeholder scale and worldwide for academics and entrepreneurs. The danger to smart grid cyber security is significantly expanding in scope as energy systems gain pervasive intelligence and communications capabilities throughout their operational processes. Numerous SCADA networks have been the target of significant cyber-attacks that badly damaged the operational control circuits and related components. In other instances, a cyber-attacker creates a knockoff by imitating the distinctive algorithmic flow embedded into the SCADA network. The internet and wireless connectivity have made it possible for hackers to quickly achieve their objectives in several industries. As a result, we proposed a GSFTNN approach with a GSF feature selection model to develop a trustworthy deep learning algorithm. Extensive experiments were conducted using the WUSTL-IIOT-2018 ICS SCADA cyber security dataset. The experimental results reveal that the proposed GSFTNN algorithm surpasses RNN, LSTM, and ResNet in both accuracy and training time. The proposed algorithm's adeptness in categorizing data and predicting outcomes expeditiously serves as a testament to its robustness. The results provide empirical evidence that the GSFTNN algorithm is a

more efficacious and efficient algorithm than the aforementioned algorithms. Performance comparison of the proposed GSFTNN model to its latest counterparts' results as in Ahakonye et al. (2023) will be investigated in the future. In addition, we will also focus on the key factors such as spectral variability in SCADA systems that could influence the model's performance. We will further improve the model's generalization ability to unfamiliar scenarios.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The data is available online.

References

Abdelkhalek, M., & Govindarasu, M. (2022). ML-based anomaly detection system for DER DNP3 communication in smart grid. In *Proc. 2022 IEEE int. conf. cyber secur. resilience, CSR 2022* (pp. 209–214). http://dx.doi.org/10.1109/CSR54599.2022.9850313.

Ahakonye, L. A. C., Nwakanma, C. I., Lee, J. M., & Kim, D.-S. (2023). Agnostic CH-DT technique for SCADA network high-dimensional data-aware intrusion detection system. *IEEE Internet of Things Journal*, 1. http://dx.doi.org/10.1109/jiot.2023.3237797.

Al Husaini, M. A. S., Habaebi, M. H., Hameed, S. A., Islam, M. R., & Gunawan, T. S. (2020). A systematic review of breast cancer detection using thermography and neural networks. *IEEE Access*, *8*, 208922–208937. http://dx.doi.org/10.1109/ACCESS.2020.3038817.

Altaha, M., Lee, J. M., Aslam, M., & Hong, S. (2020). Network intrusion detection based on deep neural networks for the SCADA system. *Journal of Physics: Conference Series*, *1585*(1), http://dx.doi.org/10.1088/1742-6596/1585/1/012038.

Altunay, H. C., Albayrak, Z., Ozalp, A. N., & Cakmak, M. (2021). Analysis of anomaly detection approaches performed through deep learning methods in SCADA systems. In *HORA 2021-3rd int. congr. human-computer interact. optim. robot. appl. proc.* http://dx.doi.org/10.1109/HORA52670.2021.9461273.

Avola, D., Cinque, L., Fagioli, A., & Foresti, G. L. (2022). SIRe-networks: Convolutional neural networks architectural extension for information preservation via skip/residual connections and interlaced auto-encoders. *Neural Networks*, *153*, 386–398. http://dx.doi.org/10.1016/j.neunet.2022.06.030.

Balla, A., Habaebi, M. H., Islam, M. R., & Mubarak, S. (2022). Applications of deep learning algorithms for supervisory control and data acquisition intrusion detection system. *Cleaner Engineering and Technology*, *9*(June), Article 100532. http://dx.doi.org/10.1016/j.clet.2022.100532.

Chen, J. I.-Z., & Chang, J.-T. (2020). Applying a 6-axis mechanical arm combine with computer vision to the research of object recognition in plane inspection. *Journal of Artificial Intelligence and Capsule Networks*, *2*(2), 77–99. http://dx.doi.org/10.36548/jaicn.2020.2.002.

Chen, R. C., Dewi, C., Huang, S. W., & Caraka, R. E. (2020). Selecting critical features for data classification based on machine learning methods. *Journal of Big Data*, *7*(1), http://dx.doi.org/10.1186/s40537-020-00327-4.

Cheng, L., Wu, X. H., & Wang, Y. (2018). Artificial flora (AF) optimization algorithm. *Applied Sciences*, *8*(3), http://dx.doi.org/10.3390/app8030329.

Cherifi, T., & Hamami, L. (2018). A practical implementation of unconditional security for the IEC 60780 — 5 — 101 SCADA protocol. *International Journal of Critical Infrastructure Protection*, *20*, 68–84. http://dx.doi.org/10.1016/j.ijcip.2017.12.001.

Gao, H., Zhang, Y., Chen, Z., Xu, S., Hong, D., & Zhang, B. (2023). A multi-depth and multi-branch network for hyperspectral target detection based on band selection. *IEEE Transactions on Geoscience and Remote Sensing*, *61*, 1. http://dx.doi.org/10.1109/tgrs.2023.3258061.

Hassan Malik, S. P., Alam, Muhammad Mahtab, Kuusik, Alar, & Moullec, Yannick Le (2020). Narrowband internet of things (NB-IoT) for industrial automation. In *Wirel. autom. as an enabler next ind. revolut* (pp. 65–87).

He, K., Zhang, X., Ren, S., & Sun, J. (2016a). Deep residual learning for image recognition. In *Proc. IEEE comput. soc. conf. comput. vis. pattern recognit., Vol. 2016-Decem* (pp. 770–778). http://dx.doi.org/10.1109/CVPR.2016.90.

He, K., Zhang, X., Ren, S., & Sun, J. (2016b). Identity mappings in deep residual networks. In *LNCS: vol. 9908, Lect. notes comput. sci. (including subser. lect. notes artif. intell. lect. notes bioinformatics)* (pp. 630–645). http://dx.doi.org/10.1007/978-3-319-46493-0_38.

Hoffmann Souza, M. L., da Costa, C. A., de Oliveira Ramos, G., & da Rosa Righi, R. (2021). A feature identification method to explain anomalies in condition monitoring. *Computers in Industry*, *133*, http://dx.doi.org/10.1016/j.compind.2021.103528.

Jasperneite, J., Sauter, T., & Wollschlaeger, M. (2020). Why we need automation models. *IEEE Industrial Electronics Magazine*, *14*(1), 29–40.

Jmila, M. I. K., & Houda (2022). Adversarial machine learning for network intrusion detection: A comparative study. *Computer Networks*, *214*(109073).

Karim, S. H., Fazle, Majumdar, Somshubra, & Darabi, Houshang (2019). Multivariate LSTM-FCNs for time series classification. *Neural Networks*, *116*, 237–245.

Khan, R. U., Zhang, X., Alazab, M., & Kumar, R. (2019). An improved convolutional neural network model for intrusion detection in networks. In *Proc. - 2019 cybersecurity cyberforensics conf. CCC 2019, No. Ccc* (pp. 74–77). http://dx.doi.org/10.1109/CCC.2019.000-6.

Khoei, T. T., Aissou, G., Hu, W. C., & Kaabouch, N. (2021). Ensemble learning methods for anomaly intrusion detection system in smart grid. In *IEEE int. conf. electro inf. technol., Vol. 2021-May* (pp. 129–135). http://dx.doi.org/10.1109/EIT51626.2021.9491891.

Kirubakaran, S. S. (2020). Study of security mechanisms to create a secure cloud in a virtual environment with the support of cloud service providers. *Journal of Trends in Computer Science and Smart Technology*, *2*(3), 148–154. http://dx.doi.org/10.36548/jtcsst.2020.3.004.

Kumar, D., & S, D. S. (2020). Enhancing security mechanisms for healthcare informatics using ubiquitous cloud. *Journal of Ubiquitous Computing and Communication Technologies*, *2*(1), 19–28. http://dx.doi.org/10.36548/jucct.2020.1.003.

Lee, J. M., & Hong, S. (2020). Keeping host sanity for security of the SCADA systems. *IEEE Access*, *8*, 62954–62968. http://dx.doi.org/10.1109/ACCESS.2020.2983179.

Liu, Q., & Wang, B. (2022). Neural extraction of multiscale essential structure for network dismantling. *Neural Networks*, *154*, 99–108. http://dx.doi.org/10.1016/j.neunet.2022.07.015.

Lopez Perez, R., Adamsky, F., Soua, R., & Engel, T. (2018). Machine learning for reliable network attack detection in SCADA systems. In *Proc. - 17th IEEE int. conf. trust. secur. priv. comput. commun. 12th IEEE int. conf. big data sci. eng. trust. 2018* (pp. 633–638). http://dx.doi.org/10.1109/TrustCom/BigDataSE.2018.00094.

Maglaras, L. A., & Jiang, J. (2014). Intrusion detection in SCADA systems using machine learning techniques. In *Proc. 2014 sci. inf. conf. SAI 2014* (pp. 626–631). http://dx.doi.org/10.1109/SAI.2014.6918252.

Mokhtari, S., Abbaspour, A., Yen, K. K., & Sargolzaei, A. (2021). A machine learning approach for anomaly detection in industrial control systems based on measurement data. *Electron*, *10*(4), 1–13. http://dx.doi.org/10.3390/electronics10040407.

Montalban, J. O. N., Iradier, E., & Member, G. S. (2020). NOMA-based 802. In *11n for industrial automation, Vol. 8*.

Ozdag, M. (2018). Adversarial attacks and defenses against deep neural networks: A survey. *Procedia Computer Science*, *140*, 152–161. http://dx.doi.org/10.1016/j.procs.2018.10.315.

P, A., Hong, J. C. D., Gao, L., Yao, J., & Zhang, B. (2020). Graph convolutional networks for hyperspectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, *59*(9), 5966–5978. http://dx.doi.org/10.1109/TGRS.2020.3015157.

Pliatsios, D., Sarigiannidis, P., Lagkas, T., & Sarigiannidis, A. G. (2020). A survey on SCADA systems: Secure protocols, incidents, threats and tactics. *IEEE Communications Surveys and Tutorials*, *22*(3), 1942–1976. http://dx.doi.org/10.1109/COMST.2020.2987688.

Rousopoulou, V., et al. (2022). Cognitive analytics platform with AI solutions for anomaly detection. *Computers in Industry*, *134*, Article 103555. http://dx.doi.org/10.1016/j.compind.2021.103555.

Samdarshi, R., Sinha, N., & Tripathi, P. (2016). A triple layer intrusion detection system for SCADA security of electric utility. In *12th IEEE int. conf. electron. energy, environ. commun. comput. control (E3-C3), INDICON 2015* (pp. 1–5). http://dx.doi.org/10.1109/INDICON.2015.7443439.

Sarker, I. H. (2022). AI-based modeling: Techniques, applications and research issues towards automation, intelligent and smart systems. *SN Computer Science*, *3*(2), 1–20. http://dx.doi.org/10.1007/s42979-022-01043-x.

Selvarajan, S., Shaik, M., Ameerjohn, S., & Kannan, S. (2020). Mining of intrusion attack in SCADA network using clustering and genetically seeded flora-based optimal classification algorithm. *IET Information Security*, *14*(1), 1–11. http://dx.doi.org/10.1049/iet-ifs.2019.0011.

Singh, V. K., Ebrahem, H., & Govindarasu, M. (2019). Security evaluation of two intrusion detection systems in smart grid SCADA environment. In *2018 north am. power symp. NAPS 2018*. http://dx.doi.org/10.1109/NAPS.2018.8600548.

Singh, P., Garg, S., Kumar, V., & Saquib, Z. (2015). A testbed for SCADA cyber security and intrusion detection. In *2015 int. conf. cyber secur. smart cities, ind. control syst. commun. SSIC 2015 - proc* (pp. 1–6). http://dx.doi.org/10.1109/SSIC.2015.7245683.

Smith, A., & Fressoli, M. (2021). Post-automation. *Futures*, *132*(June), Article 102778. http://dx.doi.org/10.1016/j.futures.2021.102778.

Teixeira, M. A., Salman, T., Zolanvari, M., Jain, R., Meskin, N., & Samaka, M. (2018). SCADA system testbed for cybersecurity research using machine learning approach. *Future Internet*, *10*(8), http://dx.doi.org/10.3390/fi10080076.

V, D. S. (2020). Automatic spotting of sceptical activity with visualization using elastic cluster for network traffic in educational campus. *Journal of Ubiquitous Computing and Communication Technologies*, *2*(2), 88–97. http://dx.doi.org/10.36548/jucct.2020.2.004.

Wang, W., Harrou, F., Bouyeddou, B., Senouci, S. M., & Sun, Y. (2022). A stacked deep learning approach to cyber-attacks detection in industrial systems: application to power system and gas pipeline systems. *Cluster Computing*, *25*(1), 561–578. http://dx.doi.org/10.1007/s10586-021-03426-w.

Wu, X., Hong, D., & Chanussot, J. (2022). Convolutional neural networks for multimodal remote sensing data classification. *IEEE Transactions on Geoscience and Remote Sensing*, *60*, http://dx.doi.org/10.1109/TGRS.2021.3124913.

Wu, X., Hong, D., & Chanussot, J. (2023). UIU-net: U-net in U-net for infrared small object detection. *IEEE Transactions on Image Processing*, *32*, 364–376. http://dx.doi.org/10.1109/TIP.2022.3228497.

Yang, H. F., & Chen, Y. P. P. (2019). Representation learning with extreme learning machines and empirical mode decomposition for wind speed forecasting methods. *Artificial Intelligence*, *277*, Article 103176. http://dx.doi.org/10.1016/j.artint.2019.103176.

Yang, H., Cheng, L., & Chuah, M. C. (2019). Deep-learning-based network intrusion detection for SCADA systems. In *2019 IEEE conf. commun. netw. secur. CNS 2019*. http://dx.doi.org/10.1109/CNS.2019.8802785.

Yang, Y., McLaughlin, K., Sezer, S., Yuan, Y. B., & Huang, W. (2014). Stateful intrusion detection for IEC 60870 − 5 − 104 SCADA security. In *IEEE power energy soc. gen. meet., Vol. 2014-Octob* (pp. 5–9). http://dx.doi.org/10.1109/PESGM.2014.6939218, no. October.