Human-centric Computing and Information Sciences

October 2025 | Volume 15

KCIA
Korea Computer Industry Association

www.hcisjournai.com

HCIS | Human-centric Computing and Information Sciences

RESEARCH (Research Manuscript) Open Access

Human-centric Computing and Information Sciences (2025) 15:58
DOI: https://doi.org/10.22967/HCIS.2025.15.058
Received: November 16, 2023; Accepted: May 26, 2024; Published: October 30, 2025

CGF-Deep-CNN: A Novel Computationally Enhanced Multiclass Cyber Attacks Detection Model for Low Powered IoT Ecosystem

Sushruta Mishra[1,*], Tarek Gaber[2,3,*], Hrudaya Kumar Tripathy[1,*], Samaresh Mishra[1], Mohammed Al-Khalidi[4], and Ali Kashif Bashir[4,5]

Abstract

Recently, heavy network traffic and significant data accumulation have been observed in smart energy-efficient wireless sensor-based applications. These power-aware sensors devices form low-power Internet of Things (IoT) ecosystem. In such applications, IoT nodes gather and analyze private data, which becomes a natural target for cyber-attacks. Many intrusion detection systems (IDSs) are designed to address this issue, but the majority of these systems are computationally expensive with high latency and fail to accurately identify subcategories of cyber-attacks. Attribute selection would help in reducing the data required for attack identification, thereby decreasing delays and memory usage for data storage, while also enhancing detection performance. In this paper, an advanced and optimized IDS model for IoT applications was proposed, utilizing a novel hybrid attribute selection method called credit gain function (CGF). This method incorporates correlation feature selection (CFS) and gain ratio. The proposed attribute selector is used to optimize the dataset through CGF, resulting in a memory-constrained dataset. By employing the proposed CFS method, a novel IDS model based on the Deep-CNN technique is recommended for detecting and classifying cyber-attacks and their sub-categories within an IoT environment. Performance analysis of the presented framework was conducted using four public datasets—IoTID20, UNSW-nb15, NSL-KDD, and KDD—under various metrics, employing different parameters for binary, multi-class, and sub-category classification. The evaluation demonstrated that the proposed IDS model is highly capable, achieving a high accuracy, precision, recall, and F-measure of 98.1%, 96.7%, 96.3%, and 96.8%, respectively. The optimal performance was attained when implementing two convolutional layers and three dense layers of the CNN model with a batch size of 64. Additionally, the presented framework was evaluated to be efficient, with a mean response delay of 2.8 seconds and a low false positive rate of 0.002%. Consequently, the proposed intrusion detection model offers a constructive solution for assessing different cyber-attacks in an IoT ecosystem.

Keywords

Cybersecurity, Internet of Things (IoT), Intrusion Detection System (IDS), Deep learning, Gain ratio, CNN, Zero-Mean Normalization

***Corresponding Author:** Hrudaya Kumar Tripathy (hktripathyfcs@kiit.ac.in), Tarek Gaber (t.m.a.gaber@salford.ac.uk)
[1]School of Computer Engineering, Kalinga Institute of Industrial Technology, Deemed to be University, Bhubaneswar, India
[2]School of Science, Engineering, and Environment, University of Salford, UK
[3]Faculty of Computers and Informatics, Suez Canal University, Ismailia, Egypt
[4]Department of Computing and Mathematics, Manchester Metropolitan University, UK
[5]Department of Computer Science & Information Technology, College of Engineering, Abu Dhabi University, Abu Dhabi, UAE

1. Introduction

With the worldwide proliferation of sensory applications, numerous cybersecurity vulnerabilities have surfaced in current Internet of Things (IoT) infrastructures. Such security vulnerabilities pose a global threat to undermine organizational structures like privacy and the mobility of functional units. These security threats impact both technological elements as well as the financial well-being of any organization [1]. Malicious attacks that attempt to exploit these vulnerabilities have brought many insecurities to digital systems and architectural infrastructures [2, 3]. IoT devices gather, store, and analyze application specific information in a distributed manner from diverse sources, making them an open target for intruders because of their decentralized nature [4]. The successful establishment of smart sensory connectivity has become heavily reliable on the security resilience of these networks [5]. Majority of these smart devices are low powered which means they are inadequately powered and possess less computation ability. Since these IoT devices are energy constrained with restricted functional power, hence the effect of a cyber-attacks may lead to catastrophic impact on the IoT ecosystem. Thus, an advanced and reliable approach for smart IoT system to identify cyber threats and ensure the security of IoT networks against intruders is an essential requirement. In such a scenario, a computationally effective network risks prediction model is needed that can process IoT network data traffic to detect and classify cyber-attacks, ultimately enhancing IoT security [6].

A generic intrusion prediction model for smart IoT setups is shown in Fig. 1. This IoT setup could exist in various application domains, such as connected healthcare, connected vehicles, supply chain management, etc. The sensory elements are monitored by a detection unit, thereby generating notifications to users or the responsive units to analyze potential security issues when the data traffic pattern is detected as a threat. The detection module needs to be trained by a suitable predictive analytics method with an appropriate classifier. Most of the misuse-oriented methods utilize pattern comparison to verify if the potential data is a threat. The above units are required to accumulate the general data patterns as well as the threat signals to build their detection framework.

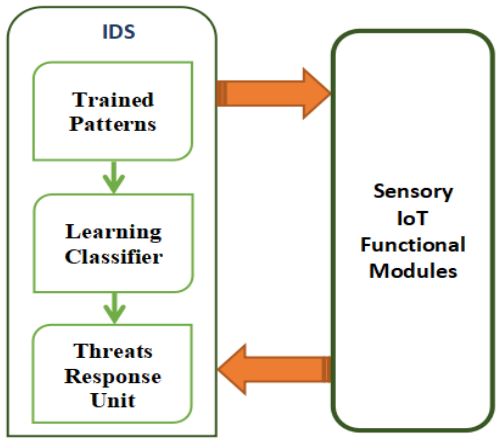

Fig. 1. A typical intrusion detection system for IoT environment.

1.1 Motivation

Several research works [7–9] have attempted to design intrusion detection system (IDS) using different predictive learning models that are able to forecast cyber-attacks with good accuracy. But in the context of IoT networks, these conventional learning methods fail to exhibit reliable performance. Existing IoT

based intrusion detection models using predictive approaches presume the devices to possess identical data patterns and packet types. But these sensory systems differ in terms of their hardware configurations, computation costs, and capability to generate different features [10]. Upon aggregation of nodes, attributes may become sparse, which impacts data modelling performance [11]. Thus, data accuracy is a real challenge in IoT systems due to their dynamic nature [12]. Here, deep learning can be considered a better alternative as it can generalize any intrusion detection problem with complex and multi-dimensional data. Although few recent works [13, 14] have used advanced deep learning models, they still face difficulty optimizing performance and identifying subcategories of cyber-attacks [15, 16].

The main limitation of the current IDS predictive approaches involving advanced prediction approaches is that they suffer from performance constraints related to resource availability, computational overhead, and dimensionality reduction, as well as failing to detect subcategories of cyber-attacks [17]. Therefore, to optimize model sparsity, redundancy, data storage, and high dimensionality in IoT systems, an efficient attribute selection method is needed for IDS systems.

1.2 Contributions

This paper introduces an advanced feature optimization method that is used for a deep learning IDS model which is able to detect and classify various cyber vulnerabilities in a heterogeneous and dynamic low powered IoT environment, as shown in Fig. 2. The paper utilizes a novel hybrid attribute selection method called credit gain function (CGF), which combines correlation feature selection (CFS) with gain ratio, thereby optimizing the scaled IoT dataset to retrieve relevant features. This CGF was then used to propose a convolutional neural network (CNN)-based IDS approach with capability to identify and classify multi-class risks for low powered IoT ecosystems. The primary contributions of this paper are highlighted as follows:

- A novel CGF attribute selection method considers credit functions of correlated attributes as well as the entropy-based gain ratio to derive an optimized dataset. This is needed because correlation coefficient methods are not suitable if attributes are not normally distributed. Also, the gain ratio by itself fails to perform well if the dataset has a high number of distinct values. In an IoT environment, there is heavy fluctuation in accumulated values, which are seldom normalized. Thus, a hybrid integration of both of the above attribute selection methods can overcome this concern. The proposed CGF attribute selection method was evaluated and compared with different attribute selection methods like info-gain, gain ratio, and chi-square methods, among others.
- A hybrid Deep-CNN based IDS which, in addition to the CGF, consists of multiple interconnected convolutional layers and dense layers to identify cyber-attacks in an IoT system. The combination helps alleviate the computational load of CNN, and has shown better performance when compared to its counterparts.
- The hybrid Deep-CNN based IDS model was evaluated using four standard datasets—IoTID20, UNSQ-nb15, NSL-KDD, and KDD—through different performance metrics including accuracy, precision, recall, F-measure, and false positive rate (FPR) for different scenarios including binary, multi-class category, and multi-class subcategory attack identification as well as latency time. The evaluation outcome demonstrates the effectiveness of the novel intrusion detection approach over other traditional existing methods. Performance evaluation validates the reliability and robustness of the deployed model, which can be used in modern smart sensory interfaces to identify malicious threats.

The overall structure of the article is presented as follows. Section 2 discusses an explanatory background overview, along with a comparative analysis of relevant works summarizing the important research gaps. The overall presented IDS model utilizing the IoT samples is detailed out in Section 3. Subsequently, the evaluation outcomes and performance analysis are given in Section 4. Finally, the paper is concluded in Section 5.

2. Literature Survey

Recently, several constructive and relevant works have been carried out that are applicable to the security issues of IoT networks. Many novel smart privacy methodologies were developed by combining machine intelligence with cybersecurity aspects [18]. Various innovative works are being undertaken for security in the IoT utilizing data analytics and advanced cognitive methods. Here, we will shed some light on existing works in the field of IDSs for the IoT using predictive learning models. Specifically, we will focus on the classification models applied, including any attribute selection methods used, and performance metrics such as the FPRs and the prediction accuracy.

Gao et al. [19] studied the recent development in IDSs, where they developed a multi-level tree algorithm. Also, an ensemble model was designed to improve the performance by using methods like random forest and deep neural networks as the base approach. The results showed that ensemble models performed better than conventional approaches. In another work by Ding and Zhai [20], the performance of conventional machine learning frameworks is analyzed. Later, they also developed a CNN model, which, upon comparison with machine learning, generated optimal performance. Though their approach exhibited a better accuracy rate than other conventional models, it should be enhanced to reduce network risk. Ingre et al. [21] presented a decision tree-enabled threat analysis model using KDD samples. CFS selected relevant attributes, and it showed the impact of this method on IDS performance.

Ever et al. [22] discussed an optimal classification method to build an IDS in their study. They applied NSL-KDD data to train their framework. In [23], the authors applied a genetic algorithm to an IoT dataset to select vital features and a support vector machine (SVM) classifier to identify the malicious packets with a recorded accuracy of 97.3%. In another piece of research undertaken in [24], the authors developed an Internetwork prediction model utilizing a variant of a neural network using NSL-KDD data on IoT systems. The outcome of the research was subdivided into binary and multi-class classifications, giving an accuracy rate of 83.28% and 81.29%, respectively. The authors in [25] proposed a CNN-based model for detection of network intrusion using IoT-collected datasets. Attribute selection methods were used to further optimize the data, and the prediction accuracy was found to be 97.7%. In [26], the authors applied a neural network to detect intrusion categories using IoT-KDD data. Principal component analysis (PCA) was the preprocessing method used to optimize the attributes, while the min-max method was used for data normalization. Feed-forward neural networks (FFNN) and Levenberg-Marquardt (LM) back propagation methods were used for classification, and the model gave around 97% accuracy. A novel deep learning-based model was developed in [27] using the NSL-KDD dataset. Label encoding with normalization was applied for preprocessing. Among all attacks predicted, the accuracy rate for denial-of-service (DoS) attacks peaked at 97%. In [28], a research analysis was carried out using machine intelligence-based intrusion detection using deep neural network (DNN) to verify the presence of any malicious attacks. Four hidden layers were used, and the ReLU function was the activation function used in the model. It recorded a maximum accuracy of around 99%.

In another work [29], the authors used the DNN model with DARPA 1999 data and used ReLU as the activation function in the hidden layer while two neurons were in the outer layer. An accuracy of 93% was determined. Basati and Faghih [30] developed a CNN-based IDS using deep attribute retrieval. It focused on IoT modules that exhibit less computational power. The developed model was validated for binary as well as multi-class labels. Rashid et al. [31] developed an ensemble-stacked technique using trees to detect intrusion in an IoT ecosystem using heterogeneous datasets. Multiple feature selectors were combined to optimize the model's performance. Fatani et al. [32] presented a new hybrid attribute reduction method for the IDS model, applying the pros of evolutionary computing. Some datasets, like BoT-IoT and CIC 2017, were used for model validation. Alkahtani and Aldhyani [33] proposed popular deep learning models like CNN, LSTM, and hybrid CNN-LSTM for intrusion prediction. The IoTID20 dataset was used for performance evaluation. Keserwani et al. [34] discussed an intrusion detection approach to extract relevant IoT network features. The technique is comprised of a combination of the evolutionary optimization method and the grey wolf optimization method. They used different datasets,

like KDDCup99 and CICIDS-2017 data. Table 1 highlights the accuracy rate analysis of some recent relevant works in context with IoT systems, where the utilization of any attribute selection is also specified [21, 23–26, 29, 35–41].

Table 1. Accuracy analysis of important existing works on intrusion detection system using IoT

Study	Attribute selection used	Classification model applied	Accuracy (%)
Ingre et al. [21]	No	Decision tree and support vector machine (SVM)	96.40
Aslahi-Shahri et al. [23]	Yes	Genetic algorithm and SVM	97.30
Yin et al. [24]	No	Recurrent neural networks	83.28
Liu et al. [25]	No	Convolutional neural network (CNN)	97.70
Singh & Ahlawat [26]	Yes	Artificial neural network (ANN) with PCA	97.97
Vigneswaran et al. [27]	No	Deep neural network (DNN) with ReLU activation function	93
Taher et al. [35]	No	ANN, SVM	94
Shah & Trivedi [36]	No	Back propagation neural network	91
Yulianto et al. [37]	No	AdaBoost	81.83
Pelletier & Abualkibash [38]	No	ANN, Random forest (RF)	96.40
Hammad et al. [39]	No	SVM, J48, RF, Zero	96.70
Faker & Dogdu [40]	No	DNN, RF	97
Amiri et al. [41]	Yes	SVM, MMIFS	86.46
CGF-Deep-CNN	Yes	DNN	98.10

Table 2. False positive rate analysis of some popular works related to IoT based IDS

Study	Model used	False positive rate (%)
Kanimozhi & Jacob [42]	Deep neural network	15
Shone et al. [45]	Stacked deep autoencoder network	2.15–14.58
Al-Zewairi et al. [43]	Deep neural network	0.56
Fu et al. [44]	Deep neural network	13.44
Farnaaz & Jabbar [46]	Random forest	0.005
Hammad et al. [39]	Decision tree	13
Pelletier & Abualkibach [38]	Neural network	7.34
CGF-Deep-CNN	Deep neural network	0.002

As highlighted in Table 1, there are many computational models used for IDS analysis in context to IoT, but very few of them implement the dimensionality reduction method. Also, as observed from the literature survey, existing models [38, 42, 43, 44] deal with classifying the cyber-attacks into binary categories. Specifically, no work is done for multi-attack classification, making it incomplete and not so scalable. Another important parameter is the FPR, which determines the number of normal events observed by the IDS as intrusions. It is vital to reduce this metric as much as possible. Some relevant works related to FPR analysis are shown in Table 2 [38, 39, 42–46], from which it can be noticed that the reviewed IoT IDS systems have a high FPR of intrusion prediction (i.e., less reliable systems) [47]. It refers to the fact that any data pattern deviating from the general trend is tagged as a cyber threat, even if it is not the scenario. This can be misleading as a result of false associations among less relevant attributes. Implementing these traditional models on the existing unbalanced IDS datasets without any preprocessing techniques or attribute selection methods results in high computational power with heavy latency delays. Further computationally advanced models using Recurrent neural networks (RNN) exhibit certain restrictions like higher computational expense due to its inability to stack up with other models, low training speed, less memory space and challenging to train model on very larger data sequence. Also multi-head attention mechanism used in transformers can be applied to deal with various

input sequence in different ways but it needs massive computational steps and it exhibits higher attribute redundancy. Hence, the existing models are less versatile and lack reliability. Thus, a more effective computationally intelligent model is needed for optimum attribute retention, optimization of response delay, and reduction of FPR using an advanced classification approach [48]. Table 3 highlights the important symbols and abbreviations used in the proposed model section.

Table 3. Abbreviations used in the proposed model

Abbreviation	Definition
BGP	Border gateway protocol
VMNET	Virtual machine networks
covr(F_i, T)	Covariance between attribute F_i and the target T
∂	Standard deviation
P	Pearson correlation coefficient
ds	Data samples
En(ds)	Entropy value for categorization
F	Attribute for which information gain is to be computed
splitInfo(attr)	Criteria for splitting of attribute
AC_{mean}	Mean attribute-class correlation value
A_{mean}	Average value of all attribute-attribute correlations
CF	Credit function of an attribute
A	Attribute count in the present subset of attributes
GV	Gain value
A_{cor}	Attributes count in the subset with optimum credit.
$A_{min}(A_{cor})$	Attribute with minimum
l	Samples present in the minor class
k	Samples present in the major class
Z''_m	Attribute value post normalization
Z'_{mn}	Zero-mean normalization
a_k	Input to convolutional layer
w_{ik}	Kernel from i to k
b_k	Output of the 1D convolutional layer
x_k	Bias of neuron in convolutional layer
r_k	Output of the max-pooling layer
$\mu(y)$	Sigmoid activation function

3. Proposed ID Model: Intrusion Detection Model for IoT Applications

IoT applications are usually subject to certain constraints, including inter-connectivity, computation capability, heterogeneous data, and energy aspects. Here, heterogeneous data are accumulated from various sensors and connected modules. Hence, a more performance-driven and computationally intelligent model that can optimize the computational power in an IoT ecosystem is required. This section explains the proposed cyber-attacks analysis model for the smart environment using an integrated preprocessing approach and a novel hybrid attribute selection method, which were then used in building a new deep learning-based IDS for malicious attack classification.

3.1 Methodology and Workflow

The proposed CGF-Deep-CNN model, as depicted in Fig. 2, aims to deal with multiple class-based cyber-attacks detection in constrained IoT systems. The model is an integrated framework comprising

various interrelated steps. After an initial preprocessing of the raw data, one-hot encoding maps the categorical features into numerical ones. A novel hybrid attribute selector (CGF) method is used to eliminate less critical attributes from the cyber-attacks dataset used in the study, thereby providing more reliable features (i.e., more reliable IDS). It not only optimizes the memory requirement but also reduces the overall latency delay incurred in implementing the model. It is followed by using the near-miss undersampling method to create a more balanced dataset. Further zero-mean normalization helps standardize the data. Then the optimized Deep-CNN technique is applied to train the model and classify the cyber-attacks.

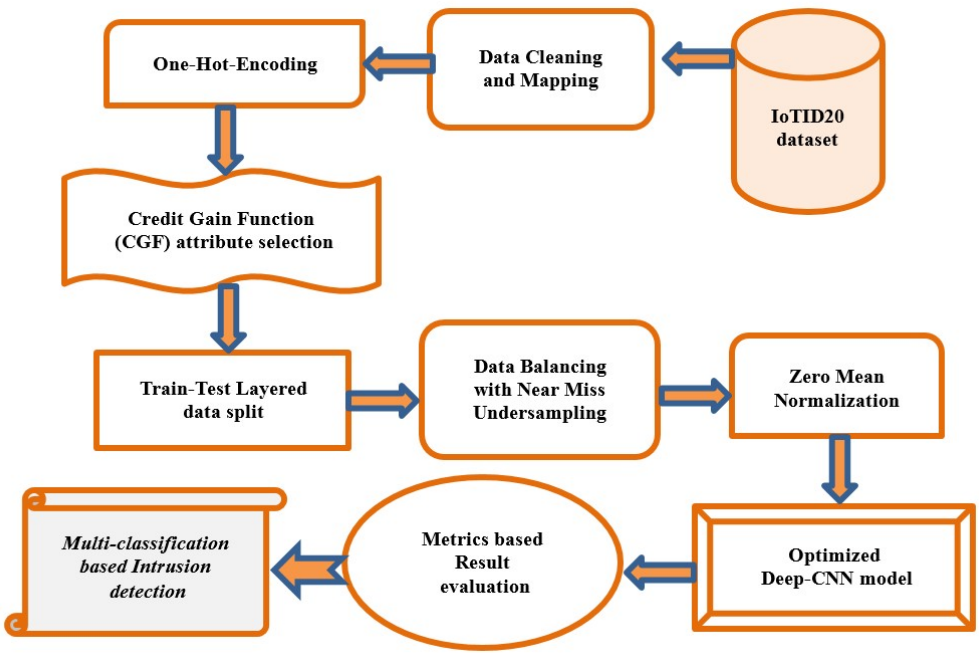

Fig. 2. The process flow of CGF-Deep-CNN model for cyber-attacks prediction in IoT.

3.2 Data Cleaning and Mapping

The raw and heterogeneous data samples of the IoT ecosystem are supposed to be cleaned at an early stage before being used for training the model. An initial preprocessing task is done to restructure the raw data into a suitable format. All possible null and undefined samples are identified and eliminated from the dataset. Also, some missing and redundant values are detected and dropped using the Python Pandas library module.

3.3 One-Hot-Encoding

To create any predictive model using machine learning or deep learning, the inputs and outputs should have integral values [33]. The IoT datasets used here possesses some categorical attributes that need to be mapped into numeric types.

One-hot-encoding is the encoding method used in this research to convert every non-numeric data value into a corresponding binary array, thereby marking the class label as 1 and others as 0. For example, nominal attributes in the NSL-KDD dataset like protocol, service and flag columns are subjected to one-hot-encoding. For protocol attribute TCP (transmission control protocol), UDP (user datagram protocol), and ICMP (Internet control message protocol) were defined. The one-hot-encoding of protocol attribute substituted the nominal data with three numeric values as shown in Table 4.

For the service attribute in NSL-KDD dataset, multiple service types were defined like "BGP," "VMNET," etc. Using one-hot-encoding process, these categorical values got replaced by their corresponding numeric values as depicted in Table 5.

Similarly, for attribute flags in NSL-KDD data, multiple flags were represented such as "OTH," "RSTR" among others. The sample example to map categorical to numeric values using one-hot-encoding is denoted in Table 6.

Also, the cyber-attacks detection accuracy is represented by analyzing each outcome against its actual value. Thus, the real values are applied for one-hot-encoding so that "1" denotes the accurate attack label and "0" represents otherwise.

Table 4. Sample of replacing categorical data of the protocol attribute with numeric values

Protocol type	TCP_Protocol	UDP_Protocol	ICMP_Protocol
TCP	1	0	0
UDP	0	1	0
ICMP	0	0	1

Table 5. Sample of replacing categorical data of the service attribute with numeric values

Service type	AOL_Service	BGP_Service	⋯	Z39_50_Service
AOL	1	0	⋯	0
BGP	0	1	⋯	0
⋯	⋯	⋯	⋯	⋯
Z39_50	0	0	⋯	1

Table 6. Sample of replacing categorical data of the flag attribute with numeric values

Flag type	OTH_flag	RSTO_flag	⋯	SF_flag
OTH	1	0	⋯	0
RSTO	0	1	⋯	0
⋯	⋯	⋯	⋯	⋯
SH	0	1	⋯	0

3.4 Novel Credit Gain Function Attribute Selection Method

Building an attribute optimized dataset is critical to improve the performance of any computational model [49] especially for an IoT enabled IDS. Redundant attributes degrade the model's reliability [47, 49]. In this study, Pearson correlation coefficient is coupled with gain ratio to form a new hybrid attribute selection method called CGF:

$$\rho = \frac{covr(F_i, T)}{\sigma(F_i \sigma_T)}. \quad (1)$$

Pearson correlation coefficient is applied when two attributes have normalized distribution. The coefficient between attribute F_i and the target T for F_1, F_2,... F_p which affects target T is denoted in Equation (1) where $covr(F_i, T)$ defines the covariance while σ is the standard deviation. It ranges between 0 to 1. Based on it, the attributes ranking is determined and the top ranked attributes on basis of 0.2 correlation coefficient are selected as shown in Equation (1).

Final attributes selection from the dataset is based on the rule set as:

An attribute F_i is selected if $\rho_{si} > 0.2$ in new dataset
Else if
An attribute F_i is rejected if $\rho_{si} < 0.2$.

Information gain ratio is the other base attribute selection method considered in this study. The information gain of each attribute is computed on basis of its entropy. The feature possessing highest information gain ratio is included in the resultant set. Entropy value for categorization is determined using Equation (2), where ds denotes the data samples.

$$En(ds) = -(P(0) * log_2(P(0)) + \left(P(0) * log_2(P(0))\right). \quad (2)$$

If an attribute F has m unique values, the dataset may be divided into m dissimilar subsets. The information gain of an attribute F is computed as shown in Equation (3), while the split information is shown in Equation (4):

$$Info - gain(ds, F) = -En(ds) - \frac{ds_i}{ds} * En(f) \quad (3)$$

$$SplitInfo(Atr) = -\sum_{i=0}^{m} \frac{ds_j}{ds} * log_2 \frac{ds_j}{ds} \quad (4)$$

The gain ratio is found by Equation (5) to normalize the information gain.

$$Gainratio = Info - \frac{gain(ds, F)}{SplitInfo(Atr)}. \quad (5)$$

Pearson correlation coefficient method is not suited if attributes are not normally distributed. Also, gain ratio fails to perform well if the dataset has a high number of distinct values. In an IoT environment, there is heavy fluctuation in accumulated values which are seldom normalized. Thus, a hybrid integration of both the above attribute selection methods can overcome this concern. So to deal with the retrieval of significant attributes in smart sensory settings, this study presents a novel integrated attribute extraction approach called CGF method to pick up more supportive attributes to detect every type of IoT cyber threats.

$$CF = \frac{A * AC_{mean}}{\sqrt{A + A(A-1)AA_{mean}}} \quad (6)$$

The credit value of the correlation coefficient method computes the most interrelated attributes. But sometimes, the selected features are absent from the final dataset. The gain value is also computed for every attribute to determine the ultimate variable set. So the least gain ratio metric of the attribute set calculated with credit value is considered to be the gain ratio threshold to pick final attributes from the raw data. Hence, along with the credit metric, the gain value is also determined to find out the final attributes. The credit metric evaluates the associations among the input features as well as the output labels. An attribute is tagged as redundant if other attributes of the dataset exhibit a high correlation with it. In Equation (6), A denotes the attribute count in the present subset, AC_{mean} represents the mean attribute-class correlation value, and A_{mean} is the average value of all attribute-attribute correlations. Using the credit function (CF), the attribute subset with the maximum credit function may be computed. In the CFS method, the selection outcome depends on the credit metric, which is a practical concern. To overcome this issue, in our CGF algorithm, the CF of each attribute is also considered.

The gain value (GV) for every attribute can be computed using Equation (7) which also helps in overcoming the overfitting issue in predictive models. Equation (8) highlights the information outcome by partitioning the data samples (DS) into the v sets, that correspond to the result on attribute A. The GV can be computed by entropy (A) – entropy (A, DS).

$$GV = \frac{gain(DS)}{split\_Info_A DS}, \quad (7)$$

$$split\_Info_A DS = -\sum_{i=1}^{v} \frac{|DS_i|}{DS} \; x \; log_2 \frac{|DS_i|}{DS}. \tag{8}$$

The pseudocode for the CGF method is shown in Pseudocode 1. In the CGF algorithm, initial attribute set is preprocessed and uploaded as the set of features as shown in Step 1. The optimum subset possessing best credit function based on higher correlation index is revealed in Step 2. Validation of the dataset is done in Step 3 to check if the optimized feature set belongs to the original features of the dataset. The least gain value of attributes is computed in Step 4. Eventually attributes with gain value more than the least gain function value and threshold are chosen as the final attributes in Step 5 and Step 6. Ultimately top attributes are noted in Step 7 and the algorithm terminates. In short, the least gain function in attribute set found utilizing credit function is applied as the threshold of gain value to pick the optimum set of attributes. Hence, the hybrid method considers both credit function as well as the gain value factors to find the optimal attribute set.

Pseudocode 1. Credit gain function (CGF)

Input:	Initial Attribute set (A_{init})
Output:	Selected Attribute set (A_{sel})
Step 1:	Preprocess and scan the initial attribute set A_{init} A_{init} = {a1, a2,…., ak}, k = initial attribute count
Step 2:	Retrieve attributes with maximum correlation as per credit function (A_{cor}) A_{cor} = {a1, a2,…..am}, m = attributes count in the subset with optimum credit.
Step 3:	Validate $A_{cor} \subseteq A_{init}$
Step 4:	Compute least Gain value of attributes in A_{cor},, $A_{min}(A_{cor})$
Step 5:	Apply equation 2 to compute gain value for all attribute of Ainit GV = {(a1, g1), (a2, g2),…..(ak, gk)}, gi is the gain value for ai ($1 \leq I \leq k$)
Step 6:	Select attributes from GV whose gain value exceeds $A_{min}(A_{cor})$
Step 7:	Output final selected attributes: A_{sel} = {s1, s2,….sf}, f = final selected attributes
Step 8:	Terminate

Applying the hybrid CGF attribute selection method in our study is a good option when used with a combined CNN model and deep learning model. As deep learning needs a large dataset, removing less relevant attributes prior to model training is better since it minimizes memory requirements and consumes less time. Also, since the deep learning algorithm is a black-box model, determining the least relevant attributes is difficult. So applying the hybrid CGF method helps uncover the most significant and least significant attributes in the dataset. This will be useful in excluding these features from future data collection and thus making the final IDS system more reliable. The complexity analysis of each step of the CGF algorithm presented in Pseudocode 1 is summarized in Table 7.

Table 7. Complexity analysis of each step of the CGF algorithm

Step	Complexity
Preprocess and scan initial attribute set	O(k)
Retrieve attributes with maximum correlation	O(m)
Validate if A_{cor} is subset of A_{init}	O(m)
Compute least gain value of attributes in A_{cor}	O(m)
Compute gain value for all attributes in A_{init}	O(k)
Select attributes with gain value exceeding $A_{min}(A_{cor})$	O(k)
Output final selected attributes	O(f)
Terminate	O(1)

k is the initial attribute count, m is the count of attributes in the subset with optimum credit (A_{cor}). Here f is the count of final selected attributes while O(1) indicates constant time complexity.

3.5 Data Balancing with Near Miss Undersampling

It has been discovered that the IoT dataset is unbalanced, which will result in incorrect recognition of the majority of instance classes. Thus, an under-sampling method can be applied to process the training set. Here near-miss under sampling (see Pseudocode 2) is used to select samples among larger class labels that have the least mean distance to the three nearest samples from less dense class labels. At first, it computes the distances between all samples from the major class and the samples from the minor class. Then it selects the k samples from the major class that exhibit the least distance from the minor class. If l samples are present in the minor class, the method will return $l * k$ samples from the major class.

Pseudocode 2. Near miss undersampling

Step 1: Computes distance between all points in major class with points in minor class.

Step 2: Select major class samples with least distance from minor class.

Step 3: Pick these k classes and store it for removal.

Step 4: Return $l * k$ samples of major class if there exists l samples of minor class

Near miss undersampling is used in this study as an effective method to handle skewed dataset. It selects instances from the majority class based on their Euclidean distance from instances in the minority class. The basic idea is to keep instances that are close to the minority class and discard instances that are far away from them. The complexity analysis of each step of the near miss undersampling algorithm presented in Pseudocode 1 is summarized in Table 8.

Table 8. Complexity analysis of each step of the near miss undersampling algorithm

Step	Complexity
Compute distance between all points	O(M*N)
Select major class samples with least distance	O(M*log(M))
Store selected samples for removal	O(K)
Return $l * k$ samples if l samples in minor class	O(K)
Compute distance between all points	O(M*N)
Select major class samples with least distance	O(M*log(M))
Store selected samples for removal	O(K)

M is the number of instances in the majority label. N is the number of instances in the minority label and K is the number of instances selected for removal and returned.

3.6 Zero-Mean Normalization

The next phase in the CGF-Deep-CNN model is the data normalization and it is needed as in the IoT dataset there exists larger differences between attribute values. Here zero-mean normalization is used to minimize the difference in varying directions for enhanced performance. The zero-mean normalization analyzes the samples by altering the mean value to 0 and the standard deviation to 1. Equation (9) is as follows:

$$Z'_{mn} = \frac{Z_{mn} - Z''_{m}}{\partial}, \tag{9}$$

where Z''_{m} and ∂ refers to the mean and standard deviation for the m-th attribute Z_m while Z'_{mn} denotes the attribute value post normalization.

3.7 Train-Test Layered Data Splitting

Data splitting into training and testing data is a vital step to determine the model's performance. Any random data partitioning may lead to unpredictable outcome. So, to prevent this, a layered approach is applied which partitions the dataset into identical homogeneous instances. In the study, the layered approach partitions the sample set into 70% training and 30% testing data for each category as shown in Table 9.

Table 9. IoTDT20 dataset division into 70:30 train and test ratio using layered splitting approach

Type	Class	Training set	Testing set
Binary	Anomaly	409739	175603
	Normal	28051	12022
Category	Mirai	290716	124593
	Scan	52685	22580
	DoS	41574	17817
	MITM ARP Spoofing	24763	10614
	Normal	28051	12022
Subcategory	Mirai-UDP Flooding	128232	54957
	Mirai-Hostbruteforceg	84825	36353
	Mirai-HTTP Flooding	39072	16746
	Mirai-Ackflooding	38586	16538
	DoS-Synflooding	41574	17817
	Scan Port OS	37151	15922
	Scan Hostport	15534	6658
	MITM ARP Spoofing	24763	10614
	Normal	28051	12022

In the study, besides using the IoTDT20 dataset, three different datasets are also applied to the model for precise identification of network risks and comparison purpose. These datasets include UNSQ-nb15, NSL-KDD, and KDD which are discussed below:

KDD dataset: This dataset was developed in 1999 and it has 41 preprocessed attributes for each network link. This dataset is segregated into four labels which include basic attributes, content attributes, time specific network attributes and host specific network attributes. It comprises a total of 4,898,430 attributes.

NSL-KDD dataset: This dataset consists of selected samples from KDD data where only relevant samples of the KDD data are included in the training data. Here the selected samples are dependent upon the proportion of samples in the KDD set. Total samples in both training and testing set are logical for which it is well suited to evaluate the complete dataset without picking small subsets.

UNSW-nb15 dataset: This dataset is consists of none network risks categories with 49 attributes along with several usual and network risks events with as many as 2,540,044 samples. A total of 221,876 samples are normal, while 321,283 samples are network risk-based records in the dataset.

3.8 Optimized Deep-CNN Model: CNN Model Integrated with DNN

Once the data is scaled and feature optimized, it can be subjected to classification. In the proposed work, a new hybrid deep learning framework is deployed that consists of a CNN model integrated with a DNN to identify cyber-attacks in smart systems, as shown in Fig. 3. Here the model is equipped with binary convolution layers and max-pooling layers, flattened with multiple dense layers. The outcome of the first convolution layer is the input to the max-pooling layer. Here, the pool size is kept at four as it helps to overcome overfitting and also create accurate sub-samples. The second convolutional layer is

operational, with the kernel size of 3 and 32 filters being utilized to generate the output. The outcome from second convolutional layer acts as the input to the max-pooling layer. In this layer, a pool size of 2 was applied and it generated the result. This combination is better suited for the model as it helps alleviate the computational load of CNN. Also, it perfectly coordinates with the data samples to build a precise activation map. The convolutional layer acts as the convergent layer for relevant attributes and also decreases the anonymous noise. The 1D convolutional layer is shown in Equations (10) and (11):

$$a_k = x_k + \sum_{i=1}^{M} (s_i, w_{ik}), \tag{10}$$

$$b_k = f(a_k), \tag{11}$$

where a_k act as the input to convolutional layer. Outcome from preceding phase is denoted as s_i, w_{ik} is the kernel from i to k. x_k is the bias of neuron in convolutional layer. The ReLU activation is denoted as $f()$. The main benefit of using ReLU activation function is that it does not activate all neurons simultaneously. Equation (12) defines the ReLU. b_k is the output of the 1D convolutional layer. Output to the convolutional layer is the input in the pooling layer demonstrated in Equation (13). The maximal value from region S containing the output values to convolutional layer is chosen. r_k is the output of the max-pooling layer.

$$f(y_k) = max(0, y_k), \tag{12}$$

$$r_k = \prod_{i \in S}^{max} z_k. \tag{13}$$

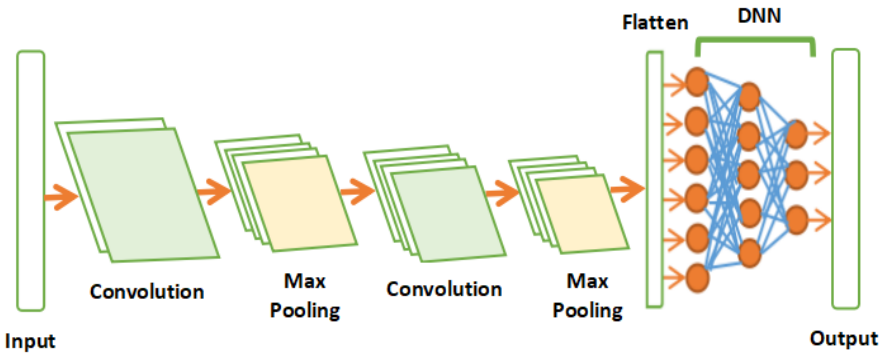

Fig. 3. Architecture of the proposed Deep-CNN model.

The flatten converts the output structure of the final pooling layer into a 1D array. Flatten output becomes the first dense layer's input. Consequentially the inputs to dense layers are processed. The ReLU activation function is utilized in dense layers. The last dense layer which generates outcome for binary classification uses sigmoid function while for multi-class category, it uses softmax function. Sigmoid and softmax are shown in Equations (14) and (15):

$$\mu(y) = \frac{1}{1 + e^{-y}} \tag{14}$$

$$softmax(y)_i = \frac{e^{y_i}}{\sum_{j=1}^{k} e^{y_j}}. \tag{15}$$

Primarily three layers are used in the model which include convolution layer, max-pooling layer, and flatten layer. The convolution layer is the main layer of the Deep-CNN model and it comprises series of

kernels. It performs linear multiplications of weights to retrieve high end features from the input data. Max-pooling is used after convolutional layer and it acts as a pooling task to compute the highest value for portions of a feature map which forms a pooled feature map. Then the flatten layer is applied to map this feature map to a template that is interpretable for dense layers. It levels the feature map to a single dimensional array.

An overview of the parameters involved in the proposed CGF-Deep-CNN model for cyber-attacks detection in IoT systems is shown in Table 10. Understanding these parameters is crucial for grasping the inner workings of the model and its potential effectiveness in detecting malicious activities.

Table 10. Discussion on parameters of the model

Model parameter	Definition
Dimensionality	The input data consists of IoT network attributes, typically represented as a 2D matrix.
Size	The input size varies depending on the dataset, with dimensions such as 80×256 for the IoTID20 dataset.
Number of convolutional layers	The model comprises multiple convolutional layers responsible for feature extraction.
Kernel size	Each convolutional layer applies a set of filters to the input data, with a specified kernel size.
Number of filters	The number of filters determines the depth of feature maps generated by each convolutional layer.
Number of pooling layers	Pooling layers are interspersed between convolutional layers to reduce spatial dimensions and control overfitting.
Pooling method	Max-pooling is commonly used to downsample feature maps by selecting the maximum value within each pool.
Number of dense layers	Dense layers follow the convolutional layers to perform classification based on extracted features.
Number of neurons	Each dense layer consists of a specified number of neurons, representing the hidden units that contribute to feature transformation.
Activation function	Activation functions such as ReLU are applied to introduce nonlinearity into the model.

4. Experimental Results and Performance Analysis

This section discusses the evaluation of the CGF selection method and the IDS detection model (CGF-Deep-CNN model). The experimental evaluation is conducted using Intel Core i7-9750H processor at 2.80 GHz and 16 GB of memory. Python was installed on the Windows 10 operating system. The suggested novel IDS model used IoTID20 data [50] for its evaluation. Both the CGF selection method and the IDS detection model (CGF-Deep-CNN model) were assessed under various scenarios: binary, multi-class categories, and multi-class subcategories classifications, as well as comparison with related methods. The performance of these scenarios was evaluated using metrics such as accuracy, precision, recall, and F-measure, as well as the FPR for the IDS detection model.

4.1 Dataset Details

The standardized data samples used here is the IoTID20 dataset, which was gathered to detect cyber-attacks in IoT systems [50]. This dataset includes advanced intercommunication data as well as new data on network interference analysis. It comprises 83 IoT network attributes with three class labels, which are binary, category, and subcategory; summary of the IoTDT20 dataset is given in Table 11.

Table 11. Cyber-attacks class information of IoTID20 dataset

Binary	Category	Subcategory
Normal	Normal	Normal
Anomaly	DoS	DoS-Synflooding
	Mirai	Mirai-Ackflooding
		Mirai-HTTP Flooding
		Mirai-Hostbruteforceg
		Mirai-UDP Flooding
	MITM	MITM ARP Spoofing
	Scan	Scan Port OS
		Scan Hostport

4.2 Result Evaluation Metrics

Validation of the proposed technique was conducted using accuracy (Ac), Precision (Pr), Recall (Re), F-measure (FM), FPR, training delay (Td), and multi-class accuracy (MCA). Certain metrics like true positive(), false negative(), false positive(), and true negative() are utilized to determine the performance values. True positive are the samples accurately detected to be general pattern. The samples that falsely classify the general pattern as a malicious threat is treated as false negatives. False positives denote cyber-attacks that are inaccurately categorized as general patterns. True negatives denote samples that are accurately categorized as potential threats. These validation parameters are represented in Equations (16)–(20).

$$Ac = \frac{\varphi + \Gamma}{\varphi + \Gamma + \Psi + \phi} \tag{16}$$

$$Pr = \frac{\varphi}{\varphi + \Psi} \tag{17}$$

$$Rc = \frac{\varphi}{\varphi + \phi} \tag{18}$$

$$FM = 2 * \frac{Pr * Rc}{Pr + Rc} \tag{19}$$

$$MCA = \frac{\omega}{\varphi + \Gamma + \Psi + \phi}. \tag{20}$$

4.3 Experimental Evaluation of Conversion of 1D Data to 2D Image Data

The process of converting 1D data into image-like data, as described in the CNN model structure, is a common technique used. This conversion allows the model to leverage CNNs, which are particularly effective in image processing tasks. In this study of anomaly detection in network traffic data, the raw data is preprocessed and mapped into image-like representations before feeding them into CNN model. This preprocessing step involves transforming the sequential data of network packets into 2D or 3D matrices, where each element represents a specific aspect of the network traffic. A major issue observed sometimes during the process is loss in information in the data. However, in this research study information loss is quite negligible since high resolution image is used in the study and also the data is normalized before being converted to image.

The values represented in Table 12 denotes the conversion of 1D data into image-like data, where for the IoTID20 dataset, each data point is converted into a 3D array to represent RGB channels, while for the other datasets, each data point is represented as a single pixel in a 2D image. As observed, there is very negligible reduction in information during this conversion process.

Table 12. Information loss analysis with network attack datasets during 1D to 2D data mapping

Dataset	Original data size	Converted image size	Information loss (%)
IoTID20	80 × 256	80 × 256 × 3	1.6
UNSW-NB15	47	47 × 1 × 1	1.4
NSL-KDD	41	41 × 1 × 1	1.4
KDD	41	41 × 1 × 1	1.5

Further, experimental evaluation of CGF method is performed with vital dimensionality reduction methods like PCA and autoencoder based methods in context to information loss. It is important to note that while PCA and CGF are linear dimensionality reduction techniques, autoencoder is a nonlinear method. Despite being linear, both PCA and CGF achieve significant reduction in information loss, making them suitable choices for dimensionality reduction tasks. However, Autoencoder, being a nonlinear method, offers even lower information loss, indicating its potential for capturing complex relationships in the data. But using autoencoder incurs heavy computational cost while it needs more time with resources like learning rate, loss function among others for model training and fine-tuning as compared to other techniques. Table 13 illustrates the comparison of information loss for different dimensionality reduction methods, including PCA, CGF, and autoencoder. The results indicate that all three methods exhibit very low information loss, with CGF and autoencoder demonstrating slightly better performance compared to PCA. But due to higher computational cost, CGF can be judged the better option.

Table 13. Information loss with network attack datasets with dimensionality reduction methods

Dataset	Information loss (%)		
	PCA	Autoencoder	CGF
IoTID20	2.2	1.3	1.6
UNSW-NB15	1.9	1.2	1.4
NSL-KDD	2.1	1.1	1.4
KDD	2.4	1.3	1.5

4.4 Experimental Analysis of Attribute Selection Methods

The novel attribute selector CGF method is evaluated against other feature selection methods. In binary classification, the CGF method outperformed other attribute selection techniques. It generated an optimum accuracy of 98.1%, and its corresponding values for precision, recall, and F-measure were 97.4%, 96.9%, and 97.2%, respectively. The reason for such promising performance is due to the fact that the CGF method removes the loosely related attributes from the dataset and it reduces the overfitting of model. The information gain and gain ratio methods also gave reasonably good performances. Fig. 4 shows the overall outcome.

As far as multi-class classification is concerned, the proposed CGF method recorded the highest accuracy of 96.8% while both info gain and gain ratio gave identical accuracy of 96.4% as observed from Fig. 5. Accordingly, the precision, recall, and F-measure with CGF method were 96.5%, 96.2%, and 96.3%, respectively.

In the scenario of multi-class sub-label classification as seen in Fig. 6, the CGF attribute selection method recorded the best performance, with 96.5%, 96.3%, 96.0%, and 96.2% being the accuracy, precision, recall, and F-measure, respectively. The gain ratio provided the next-best performance. As seen from the results, the CGF method outperforms other attribute selection methods for different class labels. The reason for such promising performance is due to the fact that the CGF method removes the loosely related attributes from the dataset and it reduces the overfitting of model.

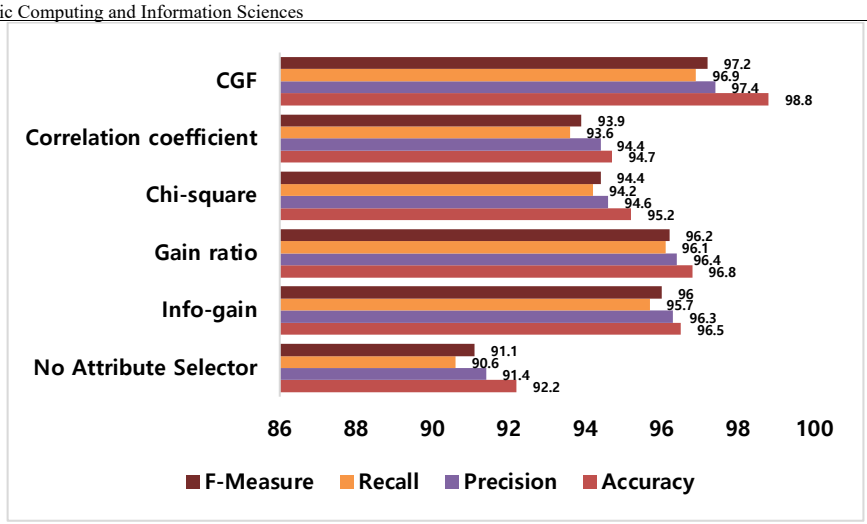

Fig. 4. Performance metrics analysis for binary class label classification using CGF method.

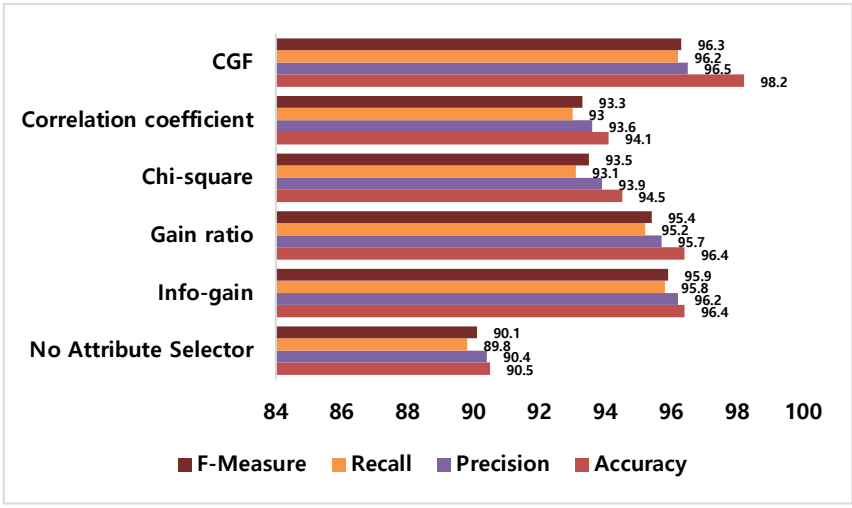

Fig. 5. Performance metrics analysis for multi-class label classification using CGF method.

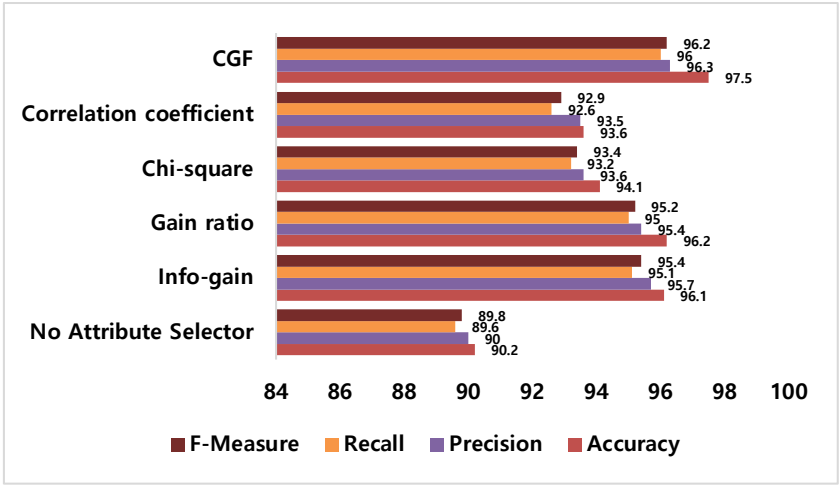

Fig. 6. Performance metrics analysis for multi-class sub-label classification using CGF method.

4.5 Experimental Analysis of Convolutional and Dense Layers of CNN Model

The validation of CGF-Deep-CNN was done for one and two convolutional layers, followed by fully connected dense I-6 layers. As seen in Table 14, for binary class label classification the performance of the proposed model with two layers of convolution layers and three dense layers was the best. With three dense layers, the proposed CGF-Deep-CNN model noted an impressive 97.4% accuracy. Similarly, the precision, recall, and F-measure values were also 97.4%, 96.9%, and 97.2%, respectively.

Table 15 depicts the performance with a multi-class classification. Again, here also, it is observed that the CGF-Deep-CNN model recorded its best performance with three dense layers and two convolutional layers. The accuracy, precision, recall, and F-measure values were noted to be 96.8%, 96.5%, 96.2%, and 96.3%, respectively.

Table 16 shows the performance with multi-class sub-label classification with multiple subcategories. Again, CGF-Deep-CNN model recorded an optimum outcome with slight decrease in metrics score. While the multi-class accuracy was recorded to be 96.5%, the corresponding precision, recall, and F-measure values were 96.3%, 96.0%, and 96.2%, respectively.

Table 14. Performance analysis of CGF-Deep-CNN with binary class label classification

	Accuracy (%)		Precision (%)		Recall (%)		F-Measure (%)	
	1-Conv	2-Conv	1-Conv	2-Conv	1-Conv	2-Conv	1-Conv	2-Conv
1-Dense	92.5	94.4	88.3	89.5	90.3	91.3	89.7	91.1
2-Dense	93.4	93.6	90.2	92.7	90.1	91.2	90.4	90.8
3-Dense	97.3	98.8	97.2	97.4	96.1	96.9	96.7	97.2
4-Dense	97.1	97.4	95.8	96.4	95.2	95.9	95.3	95.7
5-Dense	94.3	95.1	90.2	90.4	89.4	90.1	88.9	89.8
6-Dense	94.1	94.3	93.3	95.5	92.9	94.7	93.1	94.5

Table 15. Performance analysis of CGF-Deep-CNN with multi-class label classification

	Accuracy (%)		Precision (%)		Recall (%)		F-Measure (%)	
	1-Conv	2-Conv	1-Conv	2-Conv	1-Conv	2-Conv	1-Conv	2-Conv
1-Dense	92.1	94.2	87.7	88.4	90.1	90.7	89.2	90.6
2-Dense	93.1	93.2	89.7	92.2	90	90.6	89.3	90.2
3-Dense	97.3	98.2	96.1	96.5	95.6	96.2	95.5	96.3
4-Dense	96.3	96.6	95.8	96.2	94.2	95.3	94.4	95.2
5-Dense	94.1	94.6	88.1	89.5	87.6	89.4	87.8	89.3
6-Dense	93.3	93.9	92.6	94.3	90.5	93.7	92.2	93.7

Table 16. Performance analysis of CGF-Deep-CNN with multi-class sub-label classification

	Accuracy (%)		Precision (%)		Recall (%)		F-Measure (%)	
	1-Conv	2-Conv	1-Conv	2-Conv	1-Conv	2-Conv	1-Conv	2-Conv
1-Dense	92.1	94.1	87.2	88.1	90	90.3	89	90.2
2-Dense	92.4	93	89.1	91.7	88.6	90.1	89.1	89.7
3-Dense	97.3	97.5	94.9	96.3	95.2	96	95	96.2
4-Dense	95.3	96.2	95.2	95.8	94.1	95.1	94.2	95
5-Dense	93.4	94.2	87.7	89.2	87.3	89.2	87.2	89
6-Dense	93.1	93.3	92.3	94.1	90.3	93.2	91.4	93.1

4.6 Experimental Analysis of the CGF-Deep-CNN Model with Varying Batch Size

This section provides an evaluation of the CGF-Deep-CNN model with varying batch sizes for the classification of binary-class, multi-class categories, and multi-class subcategories. The batch sizes that were considered in this study are 16, 32, 64, and 128. Fig. 7 shows the analysis for binary classification, while Fig. 8 highlights the impact of CGF attribute selection on multi-class labels. The multi-class subcategory classification analysis is depicted in Fig. 9.

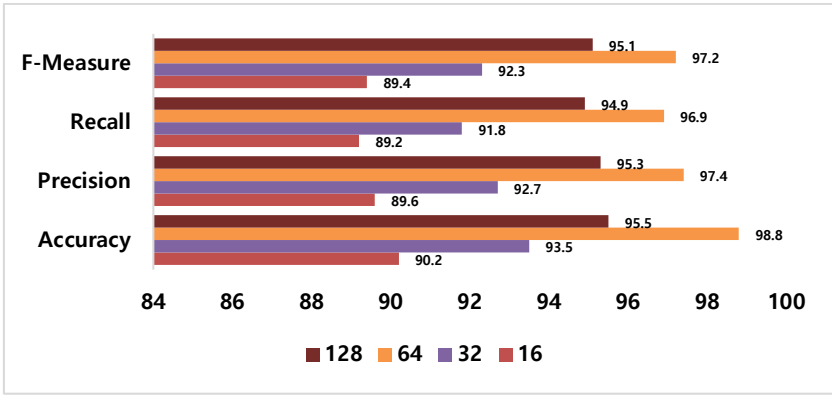

Fig. 7. Performance analysis for binary class classification using varying batch size.

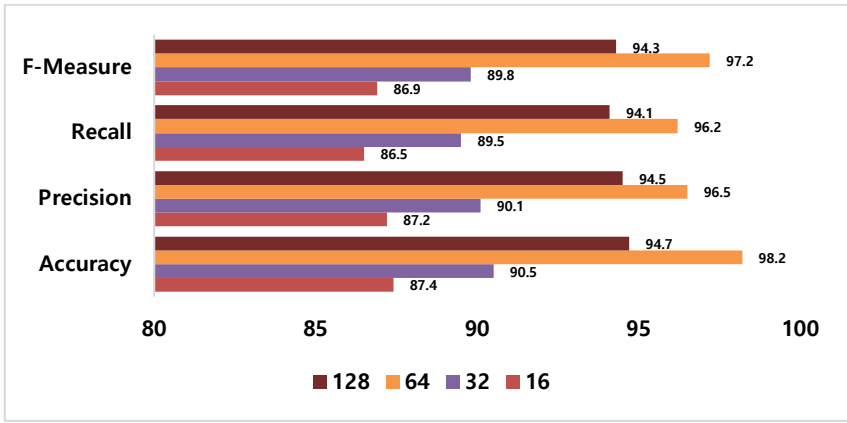

Fig. 8. Performance analysis for multi-class classification using varying batch size.

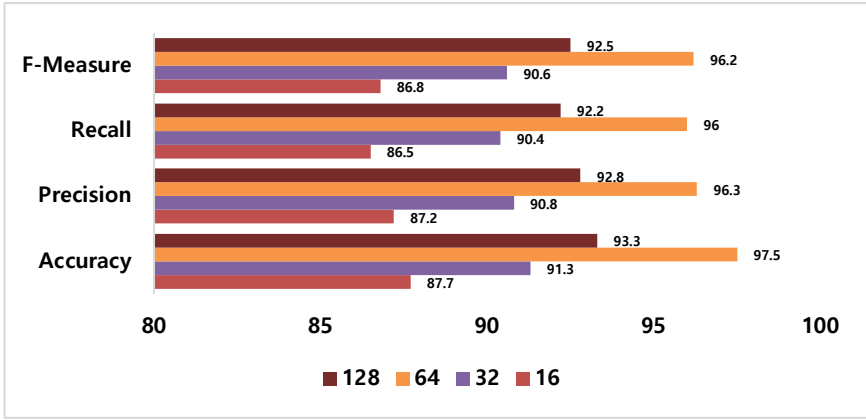

Fig. 9. Performance analysis for multi-class subcategory classification using varying batch size.

From Figs. 7–9, it can be noticed that for binary, multi-class, and multi-class subcategory classification, the developed CGF-Deep-CNN model recorded optimal performance when the batch size was 64, while the performance significantly dipped with a batch size of 16. The reason for this optimal performance is because the model is able to converge fast at a batch size of 64 and thus the error rate is least.

4.7 Comparison between CGF-Deep-CNN and Other Conventional Predictive IDS Models

In a separate experimental analysis, the CGF-Deep-CNN model was compared with the most widely used machine learning algorithms in intrusion detection (i.e., DT, LR, CNN, DBN, KNN, SVM, MLP and one-class SVM (OCSVM). Among these models, OCSVM is an effective classifier used. OCSVM is an unsupervised classification method used to learn the capability of distinguishing the testing data of a specific class form other classes. The working principle of this technique is to minimize the hypersphere of a single class of instances in the train dataset by considering all other instances external to the hypersphere to be outliers. To make this comparison fair, these machine learning techniques were also implemented and tested under the same environment settings, described in Section 4.1, and evaluated using the same IoT dataset. The analysis was again done under three main scenarios: binary, multi-class, and multi-class subcategory classifications of attacks. A summary of the results of these scenarios is given in Tables 17–19, respectively.

Table 17. Comparative analysis of traditional machine learning models with binary class classification

	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
DT	90.2	89.7	89.4	89.5
LR	93.5	93.2	93	93.1
CNN	93.5	93.1	92.6	92.8
DBN	95.5	94.8	94.2	94.5
KNN	94.2	94	93.1	93.6
OCSVM	93.9	92.4	90.5	91.2
MLP	90.3	90.1	89.4	89.7
CGF-Deep-CNN	98.8	97.4	96.9	97.2

Table 18. Comparative analysis of traditional machine learning models with multi-class classification

	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
DT	90	89.4	89	89.2
LR	92.8	92.6	92.1	92.4
CNN	93.4	93	92.4	92.6
DBN	95.1	94.5	94	94.3
KNN	93.6	93.4	92.6	92.9
OCSVM	90.6	89.8	91.1	89.9
MLP	89.8	89.6	89.2	89.5
CGF-Deep-CNN	98.2	96.5	96.2	97.2

From Tables 17–19, the following remarks can be made: Firstly, the outcome suggests the superiority of the proposed model over the others. Secondly, among other models, DBN and KNN methods gave promising results, while others did not match up to the expected performance. Thirdly, the proposed CGF-Deep-CNN model performs better than other existing methods in terms of evaluation metrics like accuracy, precision, recall and F-measure. Fourthly, among other methods, DT and SVM recorded a relatively inferior outcome, while variants of neural networks like MLP and DBN offered good results.

The CGF-Deep-CNN model recorded the optimum performance due to the use of the novel CGF method. This method acts an attribute selector which drops the less contributing features which causes overfitting of model. In IoT systems, large scale attributes are present so here CGF method eliminates the less relevant attributes while the other methods were used as a classifier.

Table 19. Comparative analysis of traditional machine learning models with multi-class subcategory classification

	Accuracy (%)	Precision (%)	Recall (%)	F-Measure (%)
DT	89.7	89.4	89	89.1
LR	92.3	91.5	91.1	91.2
CNN	92.6	92.5	92.1	92.2
DBN	94.6	94.2	93.6	93.9
KNN	93.2	93	92.4	92.6
OCSVM	90.7	91.3	89.6	90.4
MLP	89.5	89.1	88.4	88.8
CGF-Deep-CNN	97.5	96.3	96	96.2

4.8 Model Complexity Analysis

To evaluate the efficiency of the proposed model, the latency time was analyzed under the same three scenarios: binary, multi-class, and multi-class subcategory classification. The results were recorded and stored. It was found that the proposed model latency delay was minimal in all categories of attacks. While it took only 2.1 seconds with binary type, it recorded at least 2.7 seconds and 3.6 seconds in the case of multi-class and multi-class subcategory classification, respectively. Among other models, the delay was worst with SVM and MLP models. The mean response delay noted was only 2.8 seconds, and the outcome is shown in Fig. 10.

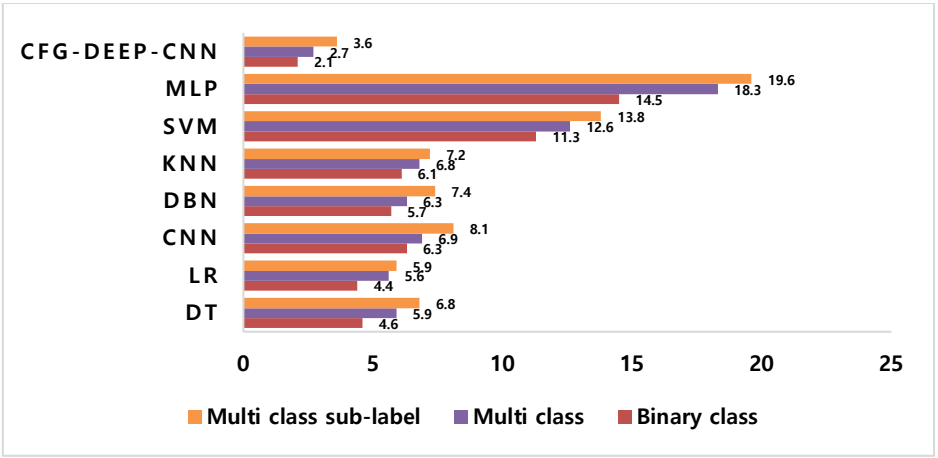

Fig. 10. Latency delay analysis of CGF-Deep-CNN model for all class labels type classification.

The overall computational complexity of the model is also evaluated in the study. The root mean square error (RMSE) value is computed for the model taking into consideration the combination of convolutional layers and dense layers. The model is implemented both using the CGF method and without using any attribute selection approach. As shown in Fig. 11, the model gave the least RMSE value of 3.216 when the CGF method is applied with 2 convolution layers and 3 dense layers.

The important parameters which impact the model's complexity is shown in Table 20. Also the vital metrics of the model with the application of CGF method is shown in Table 21. As it is observed that

with the use of CGF method for attribute selection, the model metrics are reduced by around 30%.

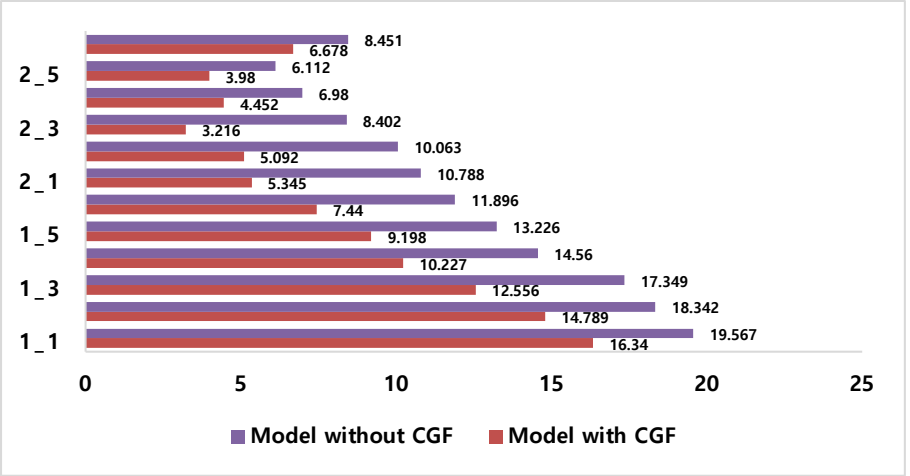

Fig. 11. Model complexity analysis in context to RMSE metric using CGF method.

Table 20. Important parameters of the Deep-CNN model

Deep-CNN model parameter	Value
Input size	[224, 224, 3]
Convolution layers	2
Convolution kernels	[3×3, 3×3]
Filters per convolutional layer	[32, 64]
Pooling layers	2
Pooling type	Max-pooling
Dense layers	2
Dense units	[128, 1]
Activation functions	ReLU, Sigmoid
Output activation	Sigmoid
Total parameters	[Approx. 500,000]

Table 21. Complexity of the model in context to parameters variables

Dataset	Input size	Conv. layers	Filters per Conv.	Pooling layers	Dense layers	CGF metrics	Deep learning model metrics
IoTID20	80 × 256	3	32	2	2	90,587	129,410
UNSW-NB15	47	2	16	1	2	976	1,394
NSL-KDD	41	3	32	2	3	3,113	4,447
KDD	41	2	16	1	2	741	1,058

4.9 Comparison with Related Works

Furthermore, a comparative analysis of the proposed CGF-Deep-CNN model is done with the existing models discussed in the literature survey section, as seen in Fig. 12. Different models defined in various existing work well in context to cyber-attack assessment. A cyber-attack detection model discussed in [23] used a genetic algorithm with SVM while [25] applied a CNN model. Both recorded very good accuracy of 97.3% and 97.7%, respectively. [38] used the AdaBoost model and generated a low accuracy of 81.83%. In comparison to the existing models, our CGF-Deep-CNN model had an impressive mean accuracy of 98.1%.

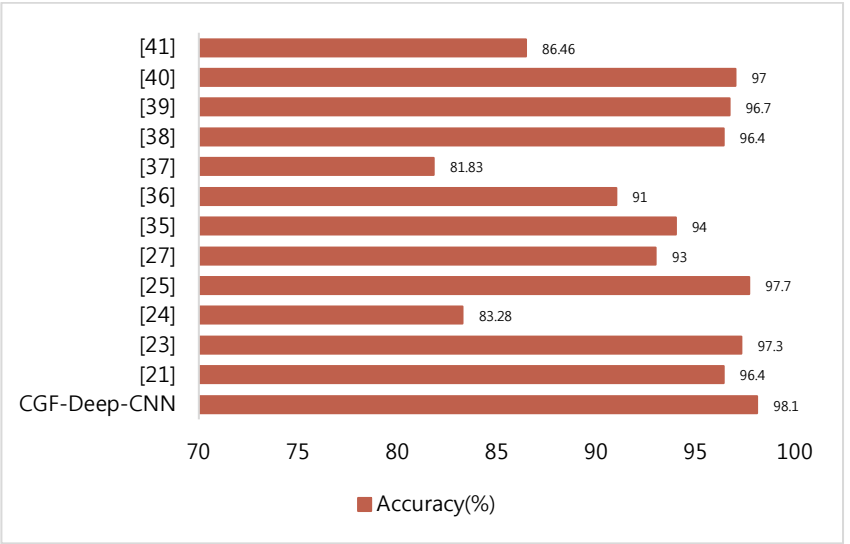

Fig. 12. Proposed model comparison with existing models in literature survey.

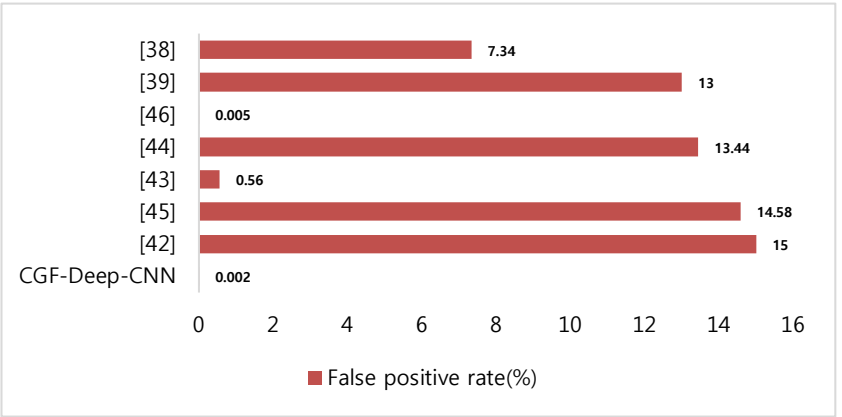

Fig. 13. False positive rate analysis of proposed model with existing models in literature survey.

As explained in [51], a reliable IDS should have a low FPR. To test the reliability of our proposed IDS model (the CGF-Deep-CNN model), we conducted a comparison of the related work. Fig. 13 highlights the impact of the FPR on the performance of the CGF-Deep-CNN model in cyber-attack detection. It is observed that the Mort-related work [45], which used deep learning techniques, had a very high FPR of 14.58%, while our model gave an optimum value of only 0.002%. This would prove that the proposed feature selection method (CGF) has helped our proposed model be more reliable. Although the work in [46] achieved a FPR close to ours, ours is still better than it. In addition, the work in [46] does not use deep learning algorithms and uses an out-dated dataset (NSL-KDD), which is not suitable for building IoT intrusion detection models. While our proposed model used a recent and more IoT-related dataset (IoTID20).

Further reliability of the proposed model using CGF attribute selection method is validated against three different datasets like UNSQ-nb15, NSL-KDD, and KDD98 to test its reliability. It is observed that the model gave consistent performance with all the considered datasets in terms of accuracy, precision, recall, and F-measure metrics. The mean accuracy, precision, recall, and F-measure values recorded were 97.35%, 95.67%, 94.37%, and 95.15%, respectively. The summary of outcome is shown in Table 22.

Table 22. Evaluation of proposed model with different datasets

	Accuracy (%)	Precision (%)	Recall (%)	F-measure (%)
IoTD20	97.8	97.4	96.9	97.2
UNSW-nb15	97.5	96.6	94	95.8
KDD98	96.9	93.5	92.2	92.8
NSL-KDD	97.2	95.2	94.4	94.8

In context to our model, both CGF method as well as the built-in feature selection of Deep-CNN model contribute to the promising performance. In an IoT based system, the incoming data flow is massive with huge number of features and to handle such huge datasets, feature selection of Deep-CNN is not so reliable. Even after extracting relevant features, some features may still get selected which affects the overall accuracy of model. Thus, the proposed CGF method is helpful in picking more significant attributes, reduces overfitting of model and helps in better model convergence. Also, it was observed that with the training and testing proportion of 70:30 ratio, the model recorded an impressive test accuracy but with 80:20, the accuracy was reduced to around 94% while with 60:40 ratio, it was around 92%. Thus, 70:30 ratio was considered for this evaluation. The proposed model achieved good convergence to optimal solution by using two methods which include Xavier initialization and Step decay. Xavier initialization was used as the initialization method for the network's weights and biases to avoid non-convergence. Step decay is the learning rate schedule used for tuning the learning rate, momentum and decay where the learning rate decreases over time.

FPR is a crucial metric considered here to validate the efficiency of the model. It denotes the proportion of data sample count wrongly categorized to be a risk and the total instance count. The FPR is particularly important in predicting intrusion attacks in an IoT system since here false alarms can be costly and disruptive. It determines the number of normal event observed by the IDS as intrusions. So the FPR should be low for a good prediction model. The proposed model successfully handles this parameter. A very low FPR of only 0.002% was recorded with the model thus generating more reliable outcome.

Thus, the outcome of the presented framework is validated and compared with other existing works. It is observed that the proposed model outperforms others in different performance metrics. Majority of existing IoT intrusion analysis models possess a high FPR to estimate intrusions. Using such models on unbalanced IDS datasets result in consuming heavy computational power and also it leads to more processing delay period. Hence the existing models are less versatile and they lack reliability. In comparison, the proposed model acts as a more effective computationally intelligent model with a novel integrated attribute selection capability with efficient preprocessing using near miss undersampling and normalization approach to optimize the response delay and reduce the FPR using an advanced classification approach. The complexity of the model was tested with different combination of convolution and dense layers with the presence of attribute selection method using RMSE metric. The model improved the prediction accuracy, reduced the FPR and it works well in different datasets thus making it more reliable.

The proposed model is developed to be deployed for both small as well as large scale networks. The use of CGF method helps in eliminating less contributing features from the large datasets generated in an IoT system. The model can also be implemented as a remote application but to accomplish it, a remote support interface (RSI) needs to be integrated to adapt to the network fluctuation issues:

- It is needed to deal with network speed fluctuations. It would acts as an alarm to enable for remote diagnosis of these fluctuations and be able to optimize network connectivity settings or even be able to suggest to user.
- It can enforce security protocols which can help in verification of the firewall and antivirus settings automatically.
- It can enable the usage of standardized applications thereby providing remote installation and upgradation functionality.

5. Conclusion

Preventing attacks on smart IoT systems and ensuring their data security have become some of the most critical needs in recent times. This study presents a new, efficient, and IoT-friendly IDS model for low-powered IoT devices. The model is built using a novel attribute selection method known as CGF, along with Deep-CNN algorithms. The proposed CGF attribute selection method was evaluated and compared with various other attribute selection methods, such as info-gain, gain ratio, and chi-square methods. The CGF method did better in terms of accuracy, precision, recall, and F-measure for both binary and multi-class category and subcategory attacks. The proposed IDS model was evaluated using four standard datasets—IoTID20, UNSW-nb15, NSL-KDD, and KDD—under different scenarios, including binary, multi-class category and subcategory classifications. It achieved a mean accuracy, precision, recall, and F-measure of 97.03%, 96.7%, 96.3%, and 96.8%, respectively. Additionally, a minimal FPR of 0.0025 was observed, enhancing the reliability of our model. Furthermore, experimental validation was conducted by varying the layers of the CNN model to determine the optimal solution. The study found that the CNN model exhibited its best performance with two convolutional layers and three dense layers. The model achieved optimal outcomes with a batch size of 64 for Deep-CNN. Evaluation demonstrated the superiority of the developed model, as it outperformed other comparative methods across the applied metrics. A latency analysis was also done. The results showed that detecting binary, multi-class, and multi-class subcategory classifications took 2.1 seconds, 2.7 seconds, and 3.6 seconds, respectively, for a mean delay of 2.8 seconds. So, it would be inferred that the designed CGF-Deep-CNN model is reliable and can be used in a real-world setting to find malicious attacks and intrusions in a smart and lightweight IoT ecosystem.

Author's Contributions

Conceptualization, SuM, HKT; Methodology, SaM, HKT, TG; Software, SM; Validation, TG, HKT, SaM; Investigation, HKT, MAK; Resources, SuM, HKT; Data curation, SuM, TG, and AKB; Writing–original draft preparation, SuM, HKT; Writing–review and editing, TG, MAK, AKS; Visualization, SuM; Supervision, SuM, TG.

Funding

None.

Competing Interests

The authors declare that they have no competing interests.

References

[1] A. A. Smadi, B. T. Ajao, B. K. Johnson, H. Lei, Y. Chakhchoukh, and O. Abu Al-Haija, "A comprehensive survey on cyber-physical smart grid testbed architectures: requirements and challenges," *Electronics*, vol. 10, no. 9, article no. 1043, 2021. https://doi.org/10.3390/electronics10091043

[2] J. H. Park, S. Yotxay, S. K. Singh, and J. H. Park, "PoAh-enabled federated learning architecture for DDoS attack detection in IoT networks," *Human-Centric Computing and Information Sciences*, vol. 14, article no. 3, 2024. https://doi.org/10.22967/HCIS.2024.14.003

[3] T. Gaber, A. El-Ghamry, and A. E. Hassanien, "Injection attack detection using machine learning for smart IoT applications," *Physical Communication*, vol. 52, article no. 101685, 2022. https://doi.org/10.1016/j.phycom.2022.101685

[4] Y. Zhang, P. Li, and X. Wang, "Intrusion detection for IoT based on improved genetic algorithm and deep belief network," *IEEE Access*, vol. 7, pp. 31711-31722, 2019. https://doi.org/10.1109/ACCESS.2019.2903723

[5] T. Gaber, J. B. Awotunde, S. O. Folorunso, S. A. Ajagbe, and E. Eldesouky, "Industrial Internet of Things intrusion detection method using machine learning and optimization techniques," *Wireless Communications and Mobile Computing*, vol. 2023, article no. 3939895, 2023. https://doi.org/10.1155/2023/3939895

[6] M. Kumar, C. Kim, Y. Son, S. K. Singh, and S. Kim, "Empowering cyberattack identification in IoHT networks with neighborhood-component-based improvised long short-term memory," *IEEE Internet of Things Journal*, vol. 11, no. 9, pp. 16638-16646, 2024. https://doi.org/10.1109/JIOT.2024.3354988

[7] A. El-Ghamry, T. Gaber, K. K. Mohammed, and A. E. Hassanien, "Optimized and efficient image-based IoT malware detection method," *Electronics*, vol. 12, no. 3, article no. 708, 2023. https://doi.org/10.3390/electronics12030708

[8] A. A. Awad, A. F. Ali, and T. Gaber, "An improved long short term memory network for intrusion detection," *PLOS One*, vol. 18, no. 8, article no. e0284795, 2023. https://doi.org/10.1371/journal.pone.0284795

[9] Q. Xia, S. Dong, and T. Peng, "An abnormal traffic detection method for IoT devices based on federated learning and depthwise separable convolutional neural networks," in *Proceedings of 2022 IEEE International Performance, Computing, and Communications Conference (IPCCC)*, Austin, TX, USA, 2022, pp. 352-359. https://doi.org/10.1109/IPCCC55026.2022.9894354

[10] K. Albulayhi and F. T. Sheldon, "An adaptive deep-ensemble anomaly-based intrusion detection system for the Internet of Things," in *Proceedings of 2021 IEEE world AI IoT congress (AIIoT)*, Seattle, WA, USA, 2021, pp. 187-196. https://doi.org/10.1109/AIIoT52608.2021.9454168

[11] H. Alrubayyi, G. Goteng, M. Jaber, and J. Kelly, "Challenges of malware detection in the IoT and a review of artificial immune system approaches," *Journal of Sensor and Actuator Networks*, vol. 10, no. 4, article no. 61, 2021. https://doi.org/10.3390/jsan10040061

[12] S. K. Singh, A. E. Azzaoui, K. K. R. Choo, L. T. Yang, and J. H. Park, "A comprehensive survey on blockchain for secure IoT-enabled smart city beyond 5G: approaches, processes, challenges, and opportunities," *Human-centric Computing and Information Sciences*, vol. 13, article no. 51, 2023. https://doi.org/10.22967/HCIS.2023.13.051

[13] S. Dong, "Multi class SVM algorithm with active learning for network traffic classification," *Expert Systems with Applications*, vol. 176, article no. 114885, 2021. https://doi.org/10.1016/j.eswa.2021.114885

[14] L. Shu, S. Dong, H. Su, and J. Huang, "Android malware detection methods based on convolutional neural network: a survey," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 7, no. 5, pp. 1330-1350, 2023. https://doi.org/10.1109/TETCI.2023.3281833

[15] S. K. Singh, M. Kumar, S. Tanwar, and J. H. Park, "GRU-based digital twin framework for data allocation and storage in IoT-enabled smart home networks," *Future Generation Computer Systems*, vol. 153, pp. 391-402, 2024. https://doi.org/10.1016/j.future.2023.12.009

[16] S. Dong, P. Wang, and K. Abbas, "A survey on deep learning and its applications," *Computer Science Review*, vol. 40, article no. 100379, 2021. https://doi.org/10.1016/j.cosrev.2021.100379

[17] Abhishek, H. K. Tripathy and S. Mishra, "A succinct analytical study of the usability of encryption methods in healthcare data security," in *Next Generation Healthcare Informatics*. Singapore: Springer, 2022, pp. 105-120. https://doi.org/10.1007/978-981-19-2416-3_7

[18] Y. Imamverdiyev and L. Sukhostat, "Anomaly detection in network traffic using extreme learning machine," in *Proceedings of 2016 IEEE 10th International Conference on Application of Information and Communication Technologies (AICT)*, Baku, Azerbaijan, 2016, pp. 1-4. https://doi.org/10.1109/ICAICT.2016.7991732

[19] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, "An adaptive ensemble machine learning model for intrusion detection," *IEEE Access*, vol. 7, pp. 82512-82521, 2019. https://doi.org/10.1109/ACCESS.2019.2923640

[20] Y. Ding and Y. Zhai, "Intrusion detection system for NSL-KDD dataset using convolutional neural networks," in *Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence*, Shenzhen, China, 2018, pp. 81-85. https://doi.org/10.1145/3297156.3297230

[21] B. Ingre, A. Yadav, and A. K. Soni, "Decision tree based intrusion detection system for NSL-KDD dataset," in *Information and Communication Technology for Intelligent Systems (ICTIS 2017) Volume 2*. Cham, Switzerland: Springer, 2017, pp. 207-218. https://doi.org/10.1007/978-3-319-63645-0_23

[22] Y. K. Ever, B. Sekeroglu, and K. Dimililer, "Classification analysis of intrusion detection on NSL-KDD using machine learning algorithms," in *Mobile Web and Intelligent Information Systems*. Cham, Switzerland: Springer, 2019, pp. 111-122. https://doi.org/10.1007/978-3-030-27192-3_9

[23] B. M. Aslahi-Shahri, R. Rahmani, M. Chizari, A. Maralani, M. Eslami, M. J. Golkar, and A. Ebrahimi, "A hybrid method consisting of GA and SVM for intrusion detection system," *Neural Computing and Applications*, vol. 27, pp. 1669-1676, 2016. https://doi.org/10.1007/s00521-015-1964-2

[24] C. Yin, Y. Zhu, J. Fei, and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, pp. 21954-21961, 2017. https://doi.org/10.1109/ACCESS.2017.2762418

[25] F. Jia and L. Z. Kong, "Intrusion detection algorithm based on convolutional neural network," *Transactions of Beijing institute of Technology*, vol. 37, no. 12, pp. 1271-1275, 2017. https://dx.doi.org/10.15918/j.tbit1001-0645.2017.12.011

[26] B. Singh and A. K. Ahlawat, "Innovative empirical approach for intrusion detection using ANN," *International Journal of Innovative Research in Computer Science & Technology (IJIRCST)*, vol. 4, no. 3, pp. 94-101, 2016. https://www.ijircst.org/view_abstract.php?year=&vol=4&primary=QVJULTI2NA==

[27] S. Potluri and C. Diedrich, "Accelerated deep neural networks for enhanced intrusion detection system," in *Proceedings of 2016 IEEE 21st International Conference on Emerging Technologies and Factory Automation (ETFA)*, Berlin, Germany, 2016, pp. 1-8. https://doi.org/10.1109/ETFA.2016.7733515

[28] J. Kim, N. Shin, S. Y. Jo, and S. H. Kim, "Method of intrusion detection using deep neural network," in *Proceedings of 2017 IEEE International Conference on Big Data and Smart Computing (BigComp)*, Jeju, South Korea, pp. 313-316. https://doi.org/10.1109/BIGCOMP.2017.7881684

[29] R. K. Vigneswaran, R. Vinayakumar, K. P. Soman, and P. Poornachandran, "Evaluating shallow and deep neural networks for network intrusion detection systems in cyber security," in *Proceedings of 2018 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT)*, Bengaluru, India, 2018, pp. 1-6. https://doi.org/10.1109/ICCCNT.2018.8494096

[30] A. Basati and M. M. Faghih, "DFE: efficient IoT network intrusion detection using deep feature extraction," *Neural Computing and Applications*, vol. 34, no. 18, pp. 15175-15195, 2022. https://doi.org/10.1007/s00521-021-06826-6

[31] M. Rashid, J. Kamruzzaman, T. Imam, S. Wibowo, and S. Gordon, "A tree-based stacking ensemble technique with feature selection for network intrusion detection," *Applied Intelligence*, vol. 52, no. 9, pp. 9768-9781, 2022. https://doi.org/10.1007/s10489-021-02968-1

[32] A. Fatani, A. Dahou, M. A. Al-Qaness, S. Lu, and M. A. Elaziz, "Advanced feature extraction and selection approach using deep learning and Aquila optimizer for IoT intrusion detection system," *Sensors*, vol. 22, no. 1, article no. 140, 2021. https://doi.org/10.3390/s22010140

[33] H. Alkahtani and T. H. Aldhyani, "Intrusion detection system to advance Internet of Things infrastructure-based deep learning algorithms," *Complexity*, vol. 2021, article no. 5579851, 2021. https://doi.org/10.1155/2021/5579851

[34] P. K. Keserwani, M. C. Govil, E. S. Pilli, and P. Govil, "A smart anomaly-based intrusion detection system for the Internet of Things (IoT) network using GWO–PSO–RF model," *Journal of Reliable Intelligent Environments*, vol. 7, no. 1, pp. 3-21, 2021. https://doi.org/10.1007/s40860-020-00126-x

[35] K. A. Taher, B. M. Y. Jisan, and M. M. Rahman, "Network intrusion detection using supervised machine learning technique with feature selection," in *Proceedings of 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST)*, Dhaka, Bangladesh, 2019, pp. 643-646. https://doi.org/10.1109/ICREST.2019.8644161

[36] B. Shah and B. H. Trivedi, "Reducing features of KDD CUP 1999 dataset for anomaly detection using back propagation neural network," in *Proceedings of 2015 5th International Conference on Advanced Computing & Communication Technologies*, Haryana, India, 2015, pp. 247-251. https://doi.org/10.1109/ACCT.2015.131

[37] A. Yulianto, P. Sukarno, and N. A. Suwastika, "Improving AdaBoost-based intrusion detection system (IDS) performance on CIC IDS 2017 dataset," *Journal of Physics: Conference Series*, vol. 1192, article no. 012018, 2019. https://doi.org/10.1088/1742-6596/1192/1/012018

[38] Z. Pelletier and M. Abualkibash, "Evaluating the CIC IDS-2017 dataset using machine learning methods and creating multiple predictive models in the statistical computing language R," *Science*, vol. 5, no. 2, pp. 187-191, 2020. https://irjaes.com/wp-content/uploads/2020/10/IRJAES-V5N2P184Y20.pdf

[39] M. Hammad, W. El-Medany, and Y. Ismail, "Intrusion detection system using feature selection with clustering and classification machine learning algorithms on the UNSW-NB15 dataset," in *Proceedings of 2020 International Conference on Innovation and Intelligence for Informatics, Computing and Technologies (3ICT)*, Sakheer, Bahrain, 2020, pp. 1-6. https://doi.org/10.1109/3ICT51146.2020.9312002

[40] O. Faker and E. Dogdu, "Intrusion detection using big data and deep learning techniques," in *Proceedings of the 2019 ACM Southeast Conference*, Kennesaw, GA, USA, 2019, pp. 86-93. https://doi.org/10.1145/3299815.3314439

[41] F. Amiri, M. R. Yousefi, C. Lucas, A. Shakery, and N. Yazdani, "Mutual information-based feature selection for intrusion detection systems," *Journal of Network and Computer Applications*, vol. 34, no. 4, pp. 1184-1199, 2011. https://doi.org/10.1016/j.jnca.2011.01.002

[42] V. Kanimozhi and P. Jacob, "UNSW-NB15 dataset feature selection and network intrusion detection using deep learning," *International Journal of Recent Technology and Engineering*, vol. 7, no. 5, pp. 443-446, 2019. https://www.ijrte.org/portfolio-item/es2080017519/

[43] M. Al-Zewairi, S. Almajali, and A. Awajan, "Experimental evaluation of a multi-layer feed-forward artificial neural network classifier for network intrusion detection system," in *Proceedings of 2017 International Conference on New Trends in Computing Sciences (ICTCS)*, Amman, Jordan, 2017, pp. 167-172. https://doi.org/10.1109/ICTCS.2017.29

[44] Y. Fu, Y. Du, Z. Cao, Q. Li, and W. Xiang, "A deep learning model for network intrusion detection with imbalanced data," *Electronics*, vol. 11, no. 6, article no. 898, 2022. https://doi.org/10.3390/electronics11060898

[45] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41-50, 2018. https://doi.org/10.1109/TETCI.2017.2772792

[46] N. Farnaaz and M. A. Jabbar, "Random forest modeling for network intrusion detection system," *Procedia Computer Science*, vol. 89, pp. 213-217, 2016. https://doi.org/10.1016/j.procs.2016.06.047

[47] A. A. Awad, A. F. Ali, and T. Gaber, "Feature selection method based on chaotic maps and butterfly optimization algorithm," in *Proceedings of the International Conference on Artificial Intelligence and Computer Vision (AICV2020)*. Cham, Switzerland: Springer, 2020, pp. 159-169. https://doi.org/10.1007/978-3-030-44289-7_16

[48] A. W. Mahmoud, R. Abdulla, M. E. Rana, and H. K. Tripathy, "IoT based energy management solution for smart green buildings," in *Proceedings of 2022 International Conference on Advancements in Smart, Secure and Intelligent Computing (ASSIC)*, Bhubaneswar, India, 2022, pp. 1-7. https://doi.org/10.1109/ASSIC55218.2022.10088306

[49] T. Gaber, A. Tharwat, V. Snasel, and A. E. Hassanien, "Plant identification: two dimensional-based vs. one dimensional-based feature extraction methods," in *10th International Conference on soft Computing Models in Industrial and Environmental Applications*. Cham, Switzerland: Springer, 2015, pp. 375-385. https://doi.org/10.1007/978-3-319-19719-7_33

[50] I. Ullah and Q. H. Mahmoud, "A scheme for generating a dataset for anomalous activity detection in IoT networks," in *Advances in Artificial Intelligence*. Cham, Switzerland: Springer, 2020, pp. 508-520. https://doi.org/10.1007/978-3-030-47358-7_52

[51] A. Guezzaz, S. Benkirane, M. Azrour, and S. Khurram, "A reliable network intrusion detection approach using decision tree with enhanced data quality," *Security and Communication Networks*, vol. 2021, article no. 1230593, 2021. https://doi.org/10.1155/2021/1230593