Cluster Computing (2024) 27:10653–10670
https://doi.org/10.1007/s10586-024-04498-0

Efficientnetv2-RegNet: an effective deep learning framework for secure SDN based IOT network

Baswaraju Swathi[1] · Soma Sekhar Kolisetty[2] · G Venkata Sivanarayana[3] · Srinivasa Rao Battula[4]

Received: 4 December 2023 / Revised: 1 April 2024 / Accepted: 1 April 2024 / Published online: 9 May 2024

Abstract
Traditional network administration required manual programming of routing policies and related parameters on specific routers and switches, which was expensive. Therefore, software-defined networking (SDN) technology has been introduced, which has boosted flexibility and decreased hardware development costs by centralizing network management. Since intrusion detection is vital in the SDN environment, this centralized architecture makes information security vulnerable to network threats. To evaluate and recognize these attacks, many researchers have recently adopted cutting-edge approaches like machine learning. However, most of these methods are not very accurate and scalable. To address this issue, this paper proposes an EfficientNetV2-RegNet-based effective deep learning technique. It effectively extracted the network features and classified the intrusions in SDN-based IoT (Internet of Things). Afterwards, an effective mitigation process was performed by a remote SDN controller to mitigate the assaults and reconfigure the network resources for trusted network hosts. Furthermore, the Conditional Generative Adversarial Network (CGAN) based data augmentation approach efficiently tackles the data imbalance issue. The most recent realistic datasets, named InSDN and IoT-23, were utilized to train and assess the presented framework to validate its efficiency. The results of the experiments demonstrated that the suggested system surpassed competitors in identifying various attack types and achieved 99.53 and 99.56% accuracy for IoT-23 and InSDN datasets, correspondingly.

Keywords EfficientNetV2 · SDN · Intrusion detection' · RegNet · Mitigation · Cycle GAN

1 Introduction

Information and communication technology have evolved significantly over the past century to change network communication patterns. The Internet of Things (IoT) is a potent network technology for communication that is revealing marvels in every aspect of our lives by bringing new notions of data transmission across networks [1–3]. This integrated IoT architecture consists of numerous incompatible electronic gadgets connected by smart sensors. However, this IoT-based network communication system demands a stable, dynamic, adaptive, faster, and more reliable design. Moreover, traditional network architecture has not changed in decades, leading to several security issues [4–6]. An innovative approach or answer to

✉ Srinivasa Rao Battula
srinivas.battula@uohyd.ac.in

Baswaraju Swathi
basawarajus@newhorizonindia.edu

Soma Sekhar Kolisetty
sekhar.soma007@gmail.com

G Venkata Sivanarayana
sgayyi@gitam.edu

[1] Department of Computer Science and Engineering (Data Science), New Horizon College of Engineering, Bengaluru, Karnataka 560103, India

[2] Department of Computer Science and Engineering, University College of Engineering Narasaraopet, JNTU, Kakinada, Andhra Pradesh 522616, India

[3] Department of Computer Science and Engineering, GST, Gitam (Deemed to Be University), Visakhapatnam, Andhra Pradesh 530045, India

[4] School of Computer and Information Sciences (SCIS), University of Hyderabad, Gachibowly, Hyderabad 500046, India

these issues is called software-defined networking (SDN), which has several characteristics that will make it the Internet's future organizational framework. The fact that this network is affordable, adaptable, and can grow without being as complicated as a conventional network is its most salient characteristic [7–9]. A controller supervises each operation in this architecture. The Open Flow protocol transmits instructions from the controller to the switches.

However, the network is vulnerable to attacks on information security due to the centralized architecture of SDN [10–12]. Gaining access to the controller could put the entire network system in danger. Attackers can utilize a variety of methods to carry out destructive actions. Therefore, this architecture needs to implement an intrusion detection system (IDS) to recognize SDN traffic [3, 13, 14]. Several approaches, including traditional IDS, antivirus software, and firewalls, have been created to safeguard systems against unauthorized access and cyber-attacks, yet they are insufficient. Due to the shortcomings of conventional IDS, machine learning and deep learning have attracted greater attention for advanced combat attacks and enhanced security [15–17]. However, machine learning-based techniques require manual feature engineering. It takes more time and effort to choose the features efficiently.

On the other hand, deep learning approaches can retrieve the features and acquire patterns from prior data to detect intrusions. Additionally, deep learning has been applied in numerous security-related tasks, including those that identify intrusions, malware, and botnets, and these applications have shown how effective deep learning models are in these fields [18, 19]. Therefore, this paper creates a deep learning-based intrusion detection framework. Moreover, most anomaly detection techniques primarily focus on identifying attacks; mitigation solutions are the subject of far less research. Furthermore, some research distributes too much aberrant traffic to different controllers for analysis, which increases the burden of other controllers and causes a delay in response, both detrimental to an assault defence. Furthermore, the datasets used by the existing systems are too old and inconsistent. The issue of unbalanced information is another important one that has rarely been addressed in previous papers. Motivated by this, an effective deep learning framework based on EfficientNetV2 and RegNet is employed in this work to detect intrusion in SDN with recent datasets. Using these techniques, the significant features of the network were extracted, and the intrusions were identified. Once the intrusions were detected, the remote SDN controller-based mitigation mechanism was activated to change the network into a normal stage. In addition to this, the data imbalance issue and overfitting were rectified using the CGAN-based approach. Moreover, the classifier's performance was improved using the MRFO-based hyperparameter tuning.

The following list highlights the proposed work's main contributions.

- Instead of using the standard feature extraction technique, EfficientNetV2-based deep learning was employed to extract the key features from the preprocessed network data.
- RegNet-based deep learning approach was used to recognize and classify different network assaults. The Manta Ray Foraging Optimization (MRFO) technique determined the classifier's parameters, reducing the classifier's learning difficulty and enabling rapid and significant accuracy categorization.
- To mitigate the network attacks, an effective mitigation process was utilized in this work to stabilize the network.
- To reduce network overfitting and solve imbalanced data problems, CGAN-based data augmentation has been implemented.

The remaining portions of the document are organized as follows. A brief overview of previous research articles is given in Sect. 2. In Sect. 3, the suggested system is deeply discussed. The dataset details and the investigation outcomes are presented in Sect. 4. Finally, Sect. 5 presents the limitations and future directions, and Sect. 6 provides the conclusion.

2 Literature review

The security weaknesses of SDN-IoT systems have been examined in several studies. In this phase, we provide a brief overview of the different approaches used to determine the threats in the SDN-IoT environment.

A deep learning approach for identifying attacks and sorting them was put forth by Ravi et al. [20]. In this paper, the in-depth characteristic illustrations of the network were retrieved using the gated recurrent unit-based deep learning layers. Then, kernel-PCA was used to choose the most beneficial attributes. The selected traits were integrated, and attack detection and classification were done using a fully connected network. Finally, the SDN-IoT dataset has been employed to evaluate this network using a range of performance metrics, and the outcomes were contrasted to other techniques.

In another work, Maray et al. [21] introduced the Convolutional Autoencoder with a Harmony Search algorithm for intrusion detection in the SDN-based IoT context. There were three main components to this system. The primary one performed the feature selection procedure by selecting important characteristics using the Harmony

Search Algorithm. Following that, the CAE approach was used to identify and classify threats. The hyperparameters were subsequently adjusted using the Artificial Fish Swarm Algorithm in the third phase. Similarly, Logeswari et al. [22] suggested HFS-LGBM IDS for SDN. Firstly, the two-phase hybrid feature selection approach minimized the data dimension and produced the best possible feature subset. The feature subset was obtained in the first stage using the Correlation-based Feature Selection technique. In the second stage, feature extraction was performed using the Random Forest Recursive Feature Elimination method. Finally, the authors used a LightGBM algorithm to identify and categorize various assault types. The NSL-KDD dataset's performance was evaluated based on the standard performance indicators.

To detect Distributed Denial-of-Service attacks, a machine learning-based framework was described by Aslam et al. [23]. The DDoS attacks were discovered by analyzing certain static aspects of the monitored network traffic using the multilayered feed-forwarded architecture based on machine learning techniques. In this architecture, the first layer used logistic regression, k-nearest neighbour, random forest, naive Bayes, and support vector machine classifiers to develop a system to recognize DDoS assaults by testing and training environment-specific datasets. The first-layer classifier's output was passed on to the Ensemble Voting algorithm, which analyzed the classifiers' performance for the final output. In order to analyze the most thorough significant elements of DDoS assaults in SDN infrastructure, El Sayed et al. [24] proposed feature selection techniques, namely Random Forest and Information Gain, to identify intrusions in SDN. Additionally, a solution based on Autoencoder and long short-term memory was devised to address the issue of DDoS intrusions in SDNs. The authors also evaluated the performance of their methods using three different datasets: CICIDS2018, CICIDS2017, and InSDN.

Elsayed et al. [25] presented a Secure Automatic Two-level IDS based on an improved Long Short-Term Memory network. It distinguishes between malicious and legitimate traffic, determines the sort of attack, and describes the high-performance sub-attack. Two of the most recent realistic datasets, ToN-IoT and InSDN, were used to train and test the recommended system to demonstrate its efficacy. They examined and contrasted its performance with that of other IDSs. For SDN, A Multi-Attack IDS was created by Ferro et al. [26]. The authors split this method down into three sections. Five of the most popular machine-learning algorithms were examined on two datasets to protect SD in the first phase to see which would work best for IoT security mechanisms. Logistic Regression, Random Forest, K-nearest neighbour, and Extreme Gradient Boosting techniques were employed in this investigation. From the investigation, the authors found that the performance of Random Forest and XGBoost was better than other algorithms. Therefore, Random Forest and XGBoost-based dual machine learning was suggested in the second phase to increase SDN-IoT security. In addition, they discovered that the CICIDS2017 dataset was better suited for performance evaluation than the NSL-KDD dataset.

Hybrid metaheuristics with deep learning enabled a model for preventing cyberattacks was created in SDN by Arun Prasad et al. [27]. First, the authors scaled the input data using data preprocessing techniques. The spiral dynamics optimization-based feature selection approach was then used to pick the most suitable featured group. In this work, assaults were identified using the hybrid recurrent neural network with a convolutional neural network model. In addition, the model's properties were modified using the Pelican Optimization Algorithm. Lastly, the Coburg Intrusion Detection Data Set was used to assess the efficacy of the recommended method. Polat et al. [28] introduced the attack detection model by combining a decision tree with a 1-dimensional convolutional neural network (1D-CNN). The model utilized 1D-CNN to obtain the online characteristics, and a decision tree-based method was employed to finalize the categorization. To develop and evaluate the procedure, this research produced a unique dataset that included many attacks on a particular research network configuration. The findings of the investigation's analysis indicate that this method can recognize breaches with a 97.8% accuracy ratio. In another study, Said et al. [29] identified network intrusions in SDN using CNN-based and BiLSTM-based techniques. The researchers combined the recursive feature reduction method with a random forest classifier to select more relevant characteristics. A random over-sampling technique was also used to balance the dataset. In the end, the CNN-BiLSTM technique was employed to conduct binary and multiclass classification, resulting in 84.23, 97.12 and 98.42% accuracy for the UNSW-NB15, InSDN and NSL-KDD datasets. Finally, the summary of this literature review is given in Table 1.

3 Proposed methodology

The Standard SDN framework contains three layers: the control, infrastructure, and application layer. The top layer is the application layer or plane that offers several services, including job scheduling, traffic management, and system security. This application plane uses the Open Flow (OF) protocol to communicate its requirements and intended network information to the control plane. Depending on the type of activity required, the control layer responds to the

requests or actions from the application layer by sending them to the network components on a specific device. The infrastructure layer contains components that regulate the network's processing and data-forwarding capacities. In this SDN framework, the proposed IDS is placed in the control layer to identify and mitigate the intrusions.

The suggested IDS's functionality is divided into five stages: preprocessing, data augmentation, feature extraction, attack detection and categorization, and mitigation. Initially, several preprocessing procedures are used to improve the quality of the raw input data. The Conditional Generative Adversarial Network (CGAN) is then used for oversampling to increase the samples of minority classes and address the issue of unbalanced data. The EfficientNetv2 and RegNet-based technique is then used to perform feature extraction and classification to recognize and classify DDoS attacks. When the attack is discovered, the remote SDN controller initiates the mitigation step. Figure 1 shows the suggested technique's system architecture.

Table 1 Summary of literature review

Study	Techniques	Dataset	Performance metrics	Drawback
Ravi et al. [20]	Gated recurrent unit with feature fusion	SDN-IoT	Precision, F1-score, recall and accuracy	This work's suggested feature fusion method is not the best one. Fusing features might not result in significant benefits if the characteristics include redundant information
Maray et al. [21]	Convolutional Autoencoder with a Harmony Search algorithm	Private dataset	Accuracy, precision, recall, F1-score, and MCC	Due to the imbalanced dataset used in this study, overfitting may occur, and the model may favour learning trends from the larger class while disregarding examples from the minority class
Logeswari et al. [22]	Random Forest Recursive Feature Elimination with LightGBM	NSL-KDD	Precision, F1-score, recall and accuracy	The dataset utilized in this paper is might not be appropriate for the SDN framework
Aslam et al. [23]	logistic regression, k-nearest neighbour, random forest, naive Bayes, support vector machine, Ensemble Voting algorithm	live.csv dataset	Precision, F1-score, recall and accuracy	This procedure involves many classifiers that may be computationally expensive. This complexity may impact systems with limited resources or real-time applications
El Sayed et al. [24]	Autoencoder and long short-term memory	CICIDS2018, CICIDS2017, and InSDN	Precision, F1-score, recall and accuracy	It can only identify attacks and non-attacks. There is no process for multiclass classification
Elsayed et al. [25]	improved Long Short-Term Memory	ToN-IoT and InSDN	Accuracy, precision, detection rate, specificity and F1-score	Large parameter sets in this network may increase the possibility of over-parameterization, which results in a system that learns only specific patterns from the training set rather than deep ones
Ferro et al. [26]	Random Forest, Logistic Regression, Extreme Gradient Boosting and K-nearest neighbor	CICIDS2017, NSL-KDD	Precision, F1-score, recall and accuracy	The dataset utilized in this paper is outdated
Arun Prasad et al. [27]	HCRNN and SDOFS	Coburg Intrusion Detection Data Set	Accuracy, precision, recall, AUC and F1-score	The intrinsic difficulty and computation necessities of HCRNN models make them challenging to implement in real-time environments, particularly those with low resource constraints
Polat et al. [28]	Random Forest and 1D-CNN	Own dataset	Accuracy, precision, specificity, recall and F1-score	This method did not use different kinds of attacks and more datasets. As a result, a model may not effectively generalize to unusual and complex attacks
Said et al. [29]	CNN and BiLSTM	InSDN, UNSW-NB15, and NSL-KDD	Precision, F1-score, recall and accuracy	The complex feature selection process and the hybrid architectures such as CNN-BiLSTM may require a longer training time. This longer training duration can make it less effective, mainly when prompt implementation of models is vital

3.1 Preprocessing phase

The data preparation phase minimized data dimensionality and removed extraneous or redundant data from the initial traffic to produce better results. To perform intrusion detection in SDN, we used two datasets. There are specific non-unique columns, such as "local_orig" and "local_resp". Therefore, these columns were removed from the dataset. Moreover, several records in the IoT-23 dataset are duplicates. Hence, they are removed from the dataset. Moreover, missing values could be found in some columns, including "duration", "orig_bytes", and "resp_bytes". These missing values were substituted using the 'Mean' of the corresponding column. The data set's categorical information was then converted to numerical form using One-hot encoding. Additionally, Min–Max normalization could be used to scale features. Equation (1) shows how each feature was scaled with this technique between 1 and 0, with each feature's highest value being transformed to 1 and its lowest value to 0.

$$S\prime = \frac{S - S_{min}}{Smin_{max}} \tag{1}$$

Here, indicates the attribute that requires normalization, while $_{\max}$ and $_{\min}$ each represent the maximum and lowest values.

3.2 CGAN-based data augmentation

To address the imbalanced data problem in the IoT23 and InSDN datasets, instances in small groups like "U2R", "Botnet", and "Torii" were raised, and some instances from big groups like "DDoS", "Probe", and "DDoS_UDP" were reduced. To do this, an augmented data set that resembles actual input data was created using the conditional GAN (CGAN) approach. In CGAN, the training procedure involved two networks: the discriminator (DR) and generator (GR) networks. These two networks competed with one another to produce signals artificially. To put it another way, the GR & DR networks engaged in the min–max game, and the following formula represents the discriminator loss (LS) function:

$$LS(DR(x), 1) = log(DR(x)) \tag{2}$$

$$LS(DR(GR(y)), 0) = log(1 - DR(GR(y))) \tag{3}$$

Here, the DR network predicts the probability of original data 'x' being denoted by DR(x), and y represents the GR network's random variable input. The final loss function of the DR network is determined by integrating the formulas mentioned above, and the discriminator's objective is to distinguish between actual and fraudulent samples accurately.

$$LS^{(DR)} = max[log(DR(x)) + log(1 - DR(GR(y)))] \tag{4}$$

On the other hand, the loss function of the generator is provided by,

$$LS^{(G)} = max[log(DR(x)) + log(1 - DR(GR(y)))] \tag{5}$$

Finally, the total value function V(GR, DR) is given in the following equation based on the entire dataset,

$$\min_{GR}\max_{DR} V(GR, DR) = \min_{GR}\max_{DR}(E_{x\sim P_{data(x)}}[\log DR(x)] + \left(E_{y\sim P_{y(y)}}[\log(1 - DR(GR(y)))]\right) \tag{6}$$

Here, the expected outcome of all original instances is represented by $E_{x\sim Pdata}(x)$, and the overall random input's expected outcome is denoted by $E_{y\sim Py}(y)$, the data distribution from the generator is shown by $P_{y(y)}$, and original data distribution is denoted by $P_{data}(x)$.

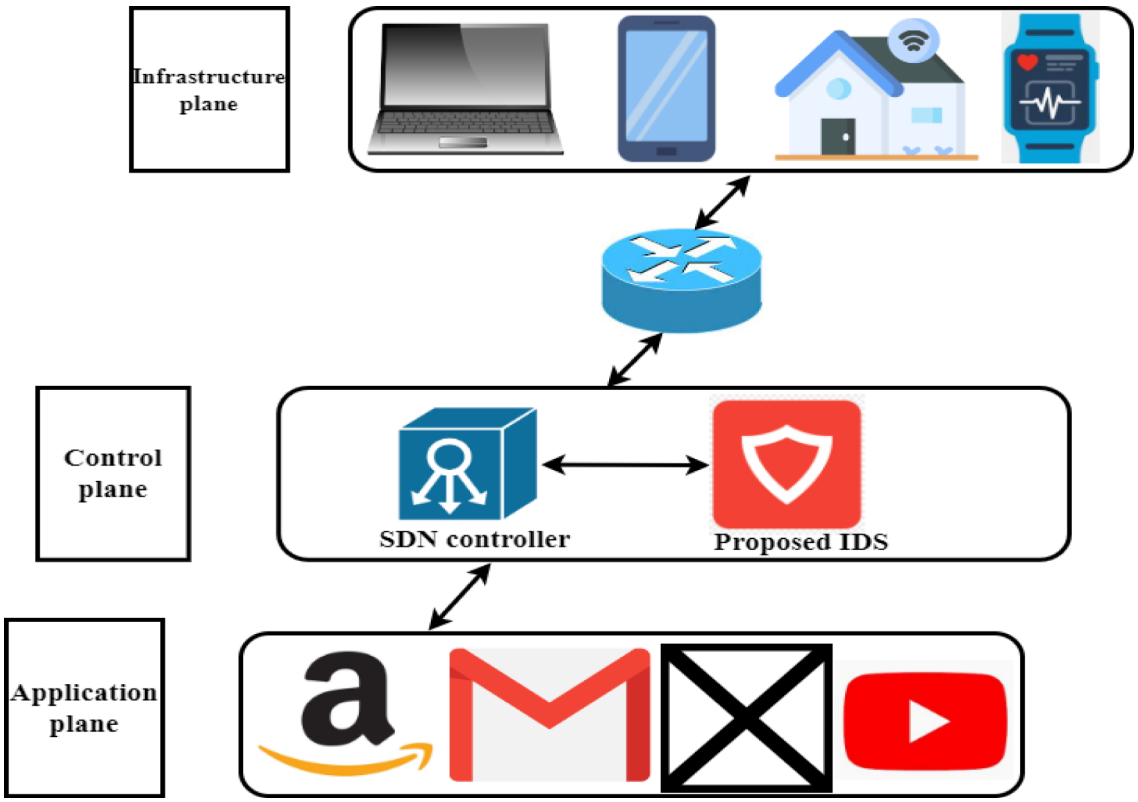
An input containing random noise was supplied into the generator network. The input feature vectors are up-sampled by the generator network's transposed convolutional layers, which then output samples with the same structure as the input data. The list of additionally generated samples is given in Appendix I.

3.3 Feature extraction using EfficientNetv2

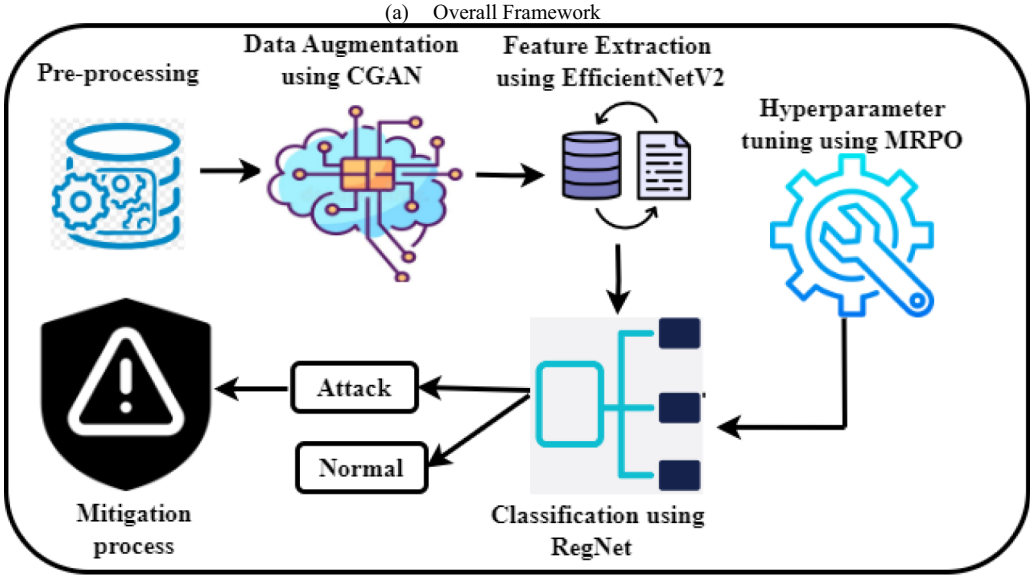
In the proposed framework, EfficientNetV2 was implemented to retrieve relevant characteristics from the input data. EfficientNetV2 is an upgraded version of EfficientNetV1, a new family of convolutional neural networks with an emphasis on two aspects: training speed and parameter efficiency. This network used a combination of compound scaling and training-aware neural architecture search. EfficientNetV2's primary building components are the fused mobile inverted bottleneck convolution (fused-MBConv) and MBConv. There are seven blocks in EfficientNetV2 networks. EfficientNetV1 only uses MBConv blocks. To save training time and simplify the model, Fused-MBConv was utilized in place of MBConv in the first three blocks. Then, 4,5,6 blocks used MBConv blocks. Block 7 comprised a conventional 1 × 1 convolution layer, a fully connected layer, and a medium pooling layer as its final layers. The MBConv block contained an SE module, a 1 × 1 ordinary convolution, a k × k depthwise convolution, a dropout layer, and a 1 × 1 ordinary convolution. Equations (7) and (8) display the equations for the convolutional layer.

$$y^l = c^{l-1} * W^l + d^l \tag{7}$$

$$c^l = f^l(y^l) \tag{8}$$



(a) Overall Framework



(b) Proposed IDS

Fig. 1 System Architecture (**a**, **b**)

Here, y^l is the result of the lth convolution, c^{l-1} is the result of the $l-1$th layer, and * is the convolution technique, W^l stands for the layer's weight, d^l for the layer's offset, c^l for the layer's result, and f^l for the layer's activation function.

The SE module comprised two completely interconnected layers plus a global average pool. The SE block used an attention strategy to enhance feature representations. Instead of ReLU as an activation function, the network used the h-swish function. The ReLU function discards values lower than zero, which could result in the loss of essential data. The h-Swish is characterized as follows:

$$h - swish(t) = t.\sigma(t) \tag{9}$$

$$\sigma(t) = \frac{ReLU6(t+3)}{6} \tag{10}$$

The piece-wise linear hard analogue function is denoted here by the symbol (t).

In this network, the fused-MBconv block was still another crucial element. A typical conv3 × 3 framework was used instead of the original MBConv framework, which included Conv1 × 1 and depthwise convolutions 3 × 3. As it passes through the layers and blocks of the network, the input data is altered and convolved. These adjustments resulted in feature maps, which were various abstract representations of the input data.

3.4 RegNet-based classification

Following feature extraction, RegNet was used to classify the extracted features. The network was built in stages using many blocks, creating a stem (the beginning), body (the middle), and head (the conclusion). The body was divided into various stages, each composed of various blocks. RegNet only employed the group convolution based standard residual bottleneck blocks.

The nine convolution layers in the suggested model consisted of three straightforward blocks with a convolution layer, followed by ReLu. The typical group convolution-based residual bottleneck block was employed in our model. A ReLU layer followed each convolution layer in a block, and each block contained a 1 × 1 convolution, a 3 × 3 group convolution, and a last 1 × 1 convolution. Using the appropriate mathematical computation, the pooling layer was often applied to the developed feature maps to reduce the number of feature maps and network parameters. We employed max-pooling with stride (s = 1) in this study. T A fully linked layer was the topmost and most significant layer in any deep learning model. The ReLU activation function was typically applied to a fully connected layer, which functioned like a multi-layer perceptron. On the other hand, the sigmoid activation function was utilized in the final layer to aid in computing the intrusion classification. Moreover, we employed a 0.5 dropout rate in the final layer to reduce overfitting.

Instead of predefined limits like depth and breadth, the design of the RegNet approach was dictated by a quantized linear formula regulated by the chosen variables. In the following equation, the block widths were calculated:

$$u_j = w_o + w_a.j for\ 0 \le j < d \tag{11}$$

Each subsequent block's width is enlarged by an element of w_a. The authors then added an extra parameter, w_0 (specified by the user), and they computed s_j:

$$w_j = w_o.w_m^{sf} \tag{12}$$

The authors quantized uj after rounding sj and computing the quantized per-block widths.

Assuming all blocks united should have the same width, all identical blocks were added to make one stage to determine each stage's breadth. To create a RegNet, the authors set the parameters g (group), b (bottleneck), wm (width parameter), wa (slope), w0 (starting width), and d (depth). The MRFO approach used the suggested algorithm to determine the values of these parameters: g = 72, b = 1, wm = 2.27, wa = 33.22, w0 = 112, and d = 25. This algorithm is explained in the following section.

3.4.1 Hyperparameter tuning using MRFO

The Manta Ray Foraging Optimization (MRFO) algorithm boosted the classifier's performance. It was influenced by the feeding strategy of a vast, wing-like marine animal known as a manta ray. This algorithm is based on how Manta rays individually look for better places and generate a population of candidate solutions. The best outcome at any given time represents the plankton, which is the subject of the emphasis. Three steps make up the search process: somersault foraging, cyclone foraging, and chain foraging.

3.4.1.1 Chain foraging phase In this phase, all fish in a group of manta rays follow the leader by travelling in a line. It is the most effective method for the chain foraging process, which is mathematically described in the following equation.

$$x_i^{t+1} = \begin{cases} xt_i + ra(x_b - x_i^t) + a(x_b - x_i^t) \to i = 1 \\ x_i^t + ra(x_{t-1}^t - x_i^t) + a(x_b - x_i^t) \to i = 2, ..., N \end{cases} \tag{13}$$

$$a = 2r\sqrt{|log(ra)|} \tag{14}$$

Here, x_i^t represents the i-th unique position at iteration (t), ra stands for the random vector corresponding to one and zero, and the best position is represented by x_b. The current location (x_i^t), the one before it (x_{t-1}^t), and the improved location (x_i^{t+1}) can all be used to implement the enhanced location.

3.4.1.2 Cyclone foraging phase The manta ray individual moves in a spiral, forming a foraging chain as it looks for food. In order to approach the target, the flocking Manta rays follow the Manta ray facing the chain and follow the spiral formation. The Manta ray's spiral motion can be mathematically described as follows in terms of how it behaves in the n-dimensional search space:

$$xt+1_i = \begin{cases} x_b + ra(x_b - x_i^t) + B.(x_b - x_i^t) \rightarrow i = 1 \\ x_b + ra(x_{i-1}^t - x_i^t) + B.(x_b - x_i^t) \rightarrow i = 2, ..., N \end{cases} \quad (15)$$

$$B = 2exp\left(ra_1.\frac{T-t+1}{T}\right).sin(2\pi ra_1), \quad (16)$$

Here, ra, ra1 ∈ [0, 1] describes a random integer, B signifies the weight factor, and T denotes the total number of iterations. Individual Manta rays can utilize the possible area and find a better option through cyclone foraging. Furthermore, assigning a random reference point allowed each person to find a novel place that was more distant from its present position. It is described as follows,

$$x_i^{t+1} = \begin{cases} X_{rand} + ra(x_{rand} - x_i^t) + B.(x_{rand} - x_i^t) \rightarrow i = 1 \\ X_{rand} + ra(x_{i-1}^t - x_i^t) + B.(x_{rand} - x_i^t) \rightarrow i = 2, ..., N \end{cases} \quad (17)$$

$$X_{rand} = l_i + ra.(u_i - l_i), \quad (18)$$

In the above equation, x_{rand} refers to a randomly chosen site arbitrarily placed and confined by the upper and lower bounds u_i and l_i.

3.4.1.3 Somersault foraging phase Every Manta ray moves independently, and it swims backwards and forwards, pivoting to enhance its location by somersaulting toward the improved position, which is obtained in the below equation:

$$x_i^{t+1} = x_i^t + \psi(ra_2 x_b - ra_3 x_i^t) \rightarrow i = 1, ..., N, \quad (19)$$

Here, ra2 and ra3 are the random values that fall between zero and one and ψ known as the somersault factor. The somersault range during the swimming of a manta ray is ψ = 2. The Manta ray can travel easily in a new domain to a better location that relies on the characteristics of somersault foraging. The range of somersaults was also proportional to the iteration because it gets smaller as the iteration increases.

3.5 Attack mitigation

When a threat is discovered, the mitigation module of our suggested structure is activated. In order to notify detection threats, the OF switches frequently inform the SDN controller with congestion rates and flow tables over the active entry and exit ports. Then, the congestion control mechanism unit is activated by the remote SDN controller to defend against attacks. The distant SDN controller instructs the running OF switches to cancel the requested connection and transfer the network assets to the authorized hosts. Then, OF switches obey the SDN controller's instructions and reject requests for data from specific source hosts, which are declared attacks. In this manner, authorized users get access to the network resources to continue the regular network activity. The pseudocode of the proposed approach is given in Algorithm 1.

Algorithm 1 Algorithm 1

Input: IoT-23 and InSDN dataset
Output: Attack categorization and mitigation

Step 1: Load and preprocess the data
Step 2: Divide the dataset into training (80%) and testing (20%)
Step 3: Apply DAGAN in the training set to augment the data.
Step 4: Use the augmented training set to train the EfficientnetV2 and Regnet
Step 5: Insert the test set after the network training.
Step 6: Employ EfficientNetV2 to carry out feature extraction
Step 7: Employ Regnet for attack detection and categorization
Step 8: Implement mitigation strategy
Step 9: Evaluate the performance of the network
Step 10: End

4 Result and discussion

This part contains the experimental evidence that illustrates how the suggested framework performs. Mininet was used to implement the suggested technique. The system used a Floodlight controller adapted to include the suggested approach. As the gateway to the IoT network, a switch with OpenFlow support is necessary. Hence, the experiment was conducted using OpenFlow 1.5. The proposed approach was implemented in the control layer and OpenFlow switch. It communicates with the gateway, receiving messages from it and responding with data.

Moreover, all the necessary libraries and packages were integrated into the Python script to generate the efficient categorization module and train the network. We divided our input data into 80–20% for training and testing. The following hyperparameter values were selected by the MRFO algorithm for deep learning network training: epochs = 100, learning rate = 0.001, weight decay rate = 1e-06, momentum = 0.8, batch size = 8 and dropout rate = 0.9. In addition to these network parameters, some specific RegNet parameters were assigned by the MRFO, which is described at the end of Sect. 3.4. Finally, the simulations of attacks demonstrated the system's resistance to different attacks in the IoT context.

4.1 Simulation metrics

The following equations are used to evaluate the experimental findings:

$$Accuracy = \frac{TRP + TRN}{TRP + TRN + FLP + FLN} \quad (20)$$

$$precision = \frac{TRP}{TRP + FLP} \quad (21)$$

$$Sensitivity = \frac{TRP}{TRP + FLN} \quad (22)$$

$$f1 - score = 2 * \frac{recall \times precision}{recall + precision} \quad (23)$$

$$specificity = \frac{TRN}{TRN + FLP} \quad (24)$$

Here, FLP (false positive) reflects the number of instances that wrongly estimated the traffic as regular traffic, FLN (false negative) indicates the quantity of instances that wrongly assumed the network as abnormal, and TRP (true positive) reflects the number of instances that accurately estimated the regular network as normal and TRN (true negative) illustrates the amount of instances that properly estimated the traffic as unusual traffic.

Additionally, precision reflects the percentage of normal traffic among all identified outcomes; accuracy indicates the ratio of accurately forecasting both normal and unusual traffic in all instances; sensitivity is comparable to detection rate, which is characterized as a measure of accurately identifying fraudulent traffic, specificity is specified as the measure of the prediction of a negative category in the dataset. F score combines Precision and Recall as an assessment.

4.2 Dataset description

InSDN dataset: The InSDN dataset includes information about a wide range of scenarios and assault categories, like Probe, DoS application, brute force, password guessing, DDoS, web, and U2R attacks. Additionally, InSDN normal traffic has several standard features. In order to imitate real attack situations, the attack sources come from both an internal and external network. CSV format contains roughly 80 statistical topics like protocol, byte quantity, packet count, and duration. The dataset includes 343,939 instances, which integrate both normal and intrusion traffic. In that, 275,515 are intrusion samples, and 68,424 are regular instances.

IoT-23 dataset: It was developed from Avast AIC lab. From 2018 to 2019, this dataset was collected in 23 different situations and released in January 2020. This comprehensive dataset includes network packets from IoT devices. The network packets were extracted to create labelled records that were malicious software-infected and clean. The total number of records in this dataset was 325,307,990 records. There were 18 features in each record, and they fell into four categories: time, content, primary, and static features. The IoT-23 dataset contained five static features: protocol, source port, destination port, and IP source and destination. Other features are determined from packets in the same flow (or in a different flow).

4.3 Evaluation results on the InSDN dataset

Numerous experiments were carried out to analyze the efficiency of the suggested IDS. The results of our initial analysis of the suggested approach's capacity for multiclass classifications are displayed in Table 2 and Fig. 2. The suggested architecture separated network assaults, according to the table, and performed above 98% for all performance measures. Particularly, the Normal, DDoS, and Prob classes achieved 99.91, 99.84, and 99.83% accuracy, respectively. The proposed strategy yielded the most negligible value in the classification of web attacks and U2R compared to other techniques because the nature of these attacks is more sophisticated and complex. However, the presented technique achieves 98.97 and 98.95% accuracy for these classes.

Following the performance of the multiclass classification, we contrasted the outcomes of the suggested strategy with those of earlier methods. Good results were attained using the InSDN dataset. Table 3 and Fig. 3 show that the suggested model has the highest intrusion detection performance in accuracy, precision, sensitivity, specificity, and f1-score (99.56, 99.48, 99.45, and 99.46%, respectively). The performance of RNN was not satisfactory when compared to existing techniques. Only 91.11% accuracy was attained by this technique. It indicates that the overall classification performance of the RNN is not satisfactory, as it has a high number of misclassifications. It concentrates more on recent facts and has trouble recalling details from the distant past. Compared to other techniques, the ILSTM approach achieves a higher accuracy (99.39%). However, this model may be complicated and prone to overfitting, which means this network works well on training data, but struggles to generalize new data. It could lead to lower precision, sensitivity, and F1-score for the model, as it might generate erroneous predictions for some instances.

Moreover, the CNN-BiLSTM technique achieved 97.12% accuracy. However, it attains a lower sensitivity value (93.67%) because the complexity of this model causes instability during training, particularly in the case of vanished or overflowing gradient difficulties. Furthermore, convolutional and recurrent layers worsen these problems, making it more challenging to train the model successfully.

The graphical representation of Table 3 is presented in Fig. 3. From Fig., we also identified that the FELIDS and LSTM achieve similar performance. However, the f1-score (85%) of FELIDS is very low compared to other approaches due to the imbalanced data issue. It reflects suboptimal performance in terms of balancing precision and recall. This issue is solved in the presented approach using a CGAN-based data augmentation technique. It balances the data between classes, reduces overfitting, and attains the best results regarding all the performance constraints.

A confusion matrix is also crucial for assessing how well a classification model works. It thoroughly explains how a model's predictions stack up against the values in a

Table 2 Multiclass classification on the InSDN dataset

Classes	Accuracy	Precision	Sensitivity	Specificity	F1-score
Normal	99.91	99.87	99.84	99.86	99.85
DDoS	99.84	99.78	99.75	99.77	99.76
DoS	99.8	99.73	99.71	99.72	99.72
Probe	99.83	99.72	99.67	99.68	99.69
Botnet	99.76	99.64	99.61	99.63	99.62
Web-attack	98.97	98.89	98.86	98.88	98.87
Password	99.49	99.41	99.38	99.4	99.39
U2R	98.95	98.84	98.81	98.83	98.82

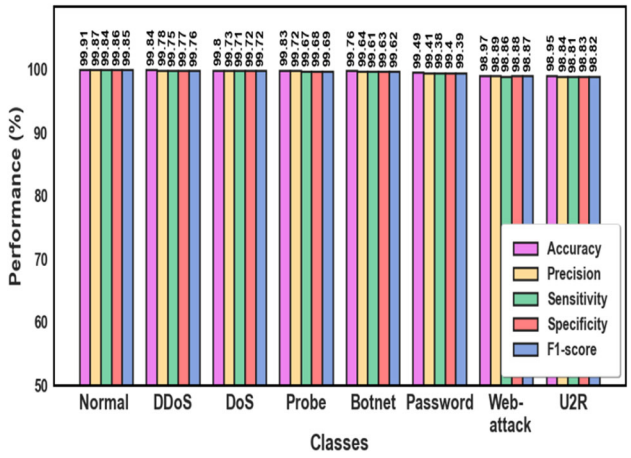


Fig. 2 Graphical evaluation of InSDN dataset

dataset that represent the ground truth. The confusion matrix from the InSDN dataset is displayed in Fig. 4. According to the matrix, the suggested approach accurately classifies most attack classes, while only a few are misclassified. The majority of misunderstandings occur between U2R and regular packets. The main reason is the slight difference between U2R and regular packets. It can be challenging to separate U2R attacks from valid administrative measures since they can display similar patterns. However, most U2R attacks are correctly classified by the proposed approach through its effective feature extraction capability.

Another performance metric named the ROC curve for the recommended method is given in Fig. 5. The ROC curve is created by changing the categorization threshold of the model while graphing the TPR (Sensitivity) on the vertical axis against the FPR on the horizontal axis. The suggested technique has demonstrated over 0.99 AUC for many intrusion classes in the InSDN dataset. It indicates that the presented strategy is consistent and can produce improved results in multiclass classification.

4.4 Evaluation results on the IoT-23 dataset

The IoT-23 dataset is utilized in this subsection to examine the ability of the suggested technique, and Table 4 and Fig. 6 show the results of the multiclass classification. It displays how well the classification model has performed, with high F1 scores, accuracy, precision, sensitivity, and specificity (above 99%) for each class. These metrics show the model to be robust in minimizing both false positives and false negatives and successful at accurately categorizing cases for all classes. It is a clear sign that a multiclass classification model is working correctly.

With the help of the presented identification system, we can distinguish between different attack classes and decide if incoming communication is legitimate or malicious. An obvious benefit of distinctly identifying every intrusion is to improve the mitigation strategy's capacity to prevent only the particular kind of traffic causing the assault rather than all sorts of traffic heading toward the victim(s).

Table 3 Comparative Analysis of the InSDN Dataset

Techniques	Accuracy	Precision	Sensitivity	Specificity	F1-score
FELIDS [30]	97.84	80.5	92.1	–	85
LSTM [31]	96.32	97.60	97.24		97.42
RFLSTM [32]	99	99	99	–	98
RNN [33]	91.11	89.94	99.37	–	94.51
CNN [34]	93.01	93.74	92.81	–	92.95
ILSTM [25]	99.39	98.5	98.4	–	98.45
CNN-BILSTM [29]	97.12	96.12	93.67	–	94.35
Proposed	99.56	99.48	99.45	99.47	99.46

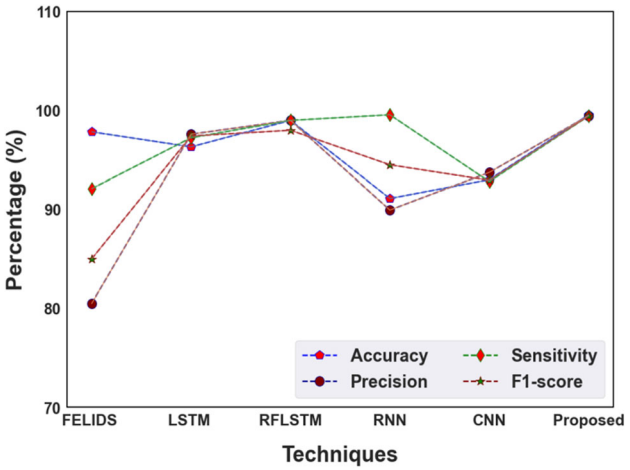


Fig. 3 Comparison of performance metrics on the InSDN dataset

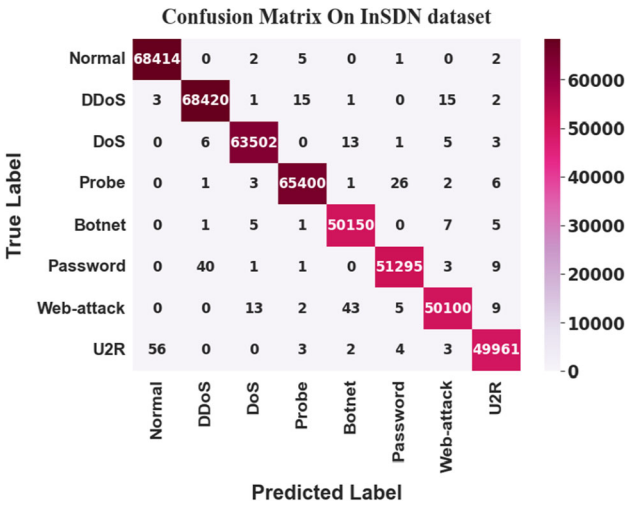


Fig. 4 InSDN dataset's confusion matrix

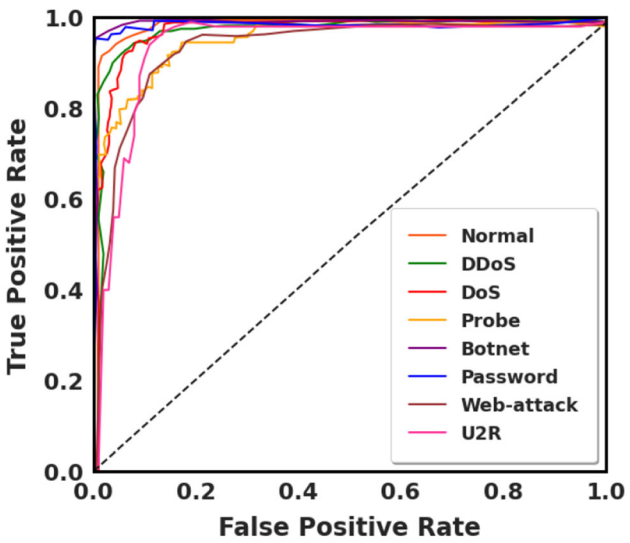


Fig. 5 ROC curve for InSDN dataset

Despite having an identification mechanism, we retrieved features separately for each host that received incoming traffic for some time. As a result, we can distinguish between servers with regular traffic and those receiving assault traffic. In order to prevent traffic for a specific host in the event of an attack, the controller can implement flow rules inside the switches.

Furthermore, in this work, we compared the performance of our proposed model with five current techniques: CNN-LSTM, HSAS-MD, BiGAN + KNN, MM-WMVEDL, AdaBoost, and MM-WMVEDL. For testing, all of these models were trained using the IoT-23 data set, and the results are shown in Table 5 and Fig. 7. The table demonstrates that the suggested model outperforms existing ones, reaching 99.53% accuracy, 99.49% precision, 99.45% sensitivity, 99.48% specificity, and 99.46% f1-score.

Compared with existing methods, the performance of Adaboost is not up to the mark, and it attains 87% accuracy, 86% precision, 87% sensitivity, and 83% f1-score only. This method is susceptible to erratic data. Therefore, effective preprocessing techniques are required to acquire better results. We perform many effective preprocessing procedures using the presented technique to make the data more compatible and achieve superior results. Furthermore, the DNN model achieved lower sensitivity (92%). Variations in input data, such as adjustments to attack tactics, traffic patterns, or network protocols, may cause DNNs to become sensitive. Challenging instances can exploit this sensitivity, deliberately constructed inputs intended to fool the model and compromise its efficacy.

Moreover, HSAS-MD and CNN-LSTM techniques attain similar results (overall 95%), not more than the proposed approach (99%). Furthermore, these techniques contain many parameters that require proper tuning. The proposed framework's MRFO algorithm performed efficient hyperparameter tuning and improved the classifier's learning ability.

We also show the confusion matrix to explain the excellent performance of our model's categorization. The confusion matrix provides a summary of the accurate and incorrect predictions. Figure 8 presents the confusion matrix of the IoT-23 dataset. The non-diagonal values in that matrix represent mistakenly anticipated samples, while the diagonal values represent successfully predicted samples. The classifier algorithm had a higher performance from the image, with the least amount of false positives and false negatives. Furthermore, the majority of the data set's samples made accurate predictions.

When we need to know how well the model can distinguish between positive and negative instances at various threshold levels, the ROC curve is a valuable tool for evaluating the performance of multiclass classification

Table 4 Multiclass categorization result of IoT-23 dataset

Classes	Accuracy	Precision	Sensitivity	Specificity	F1-score
Benign	99.82	99.77	99.74	99.76	99.75
DDoS	99.77	99.71	99.68	99.7	99.69
Mirai	99.67	99.64	99.61	99.63	99.62
Okiru	99.63	99.6	99.57	99.59	99.58
Torii	99.55	99.47	99.43	99.46	99.44
PartofHorizontalPortScan	99.35	99.32	99.29	99.31	99.3
Heartbeat	99.23	99.2	99.17	99.19	99.18
FileDownload	99.42	99.38	99.34	99.37	99.35
Command and control	99.35	99.34	99.3	99.33	99.31

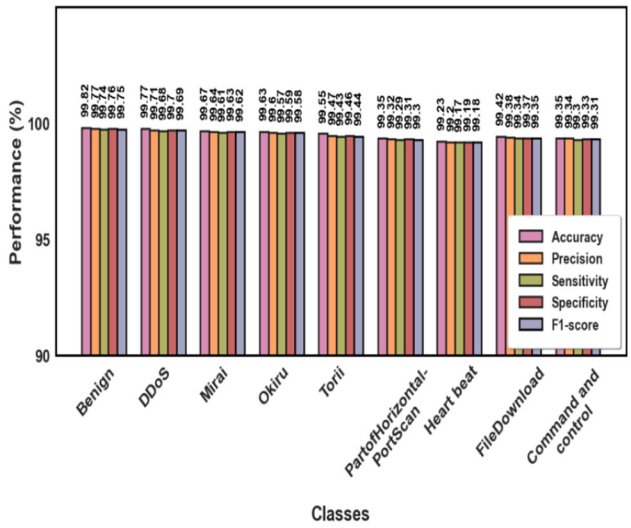


Fig. 6 Graphical representation of multiclass classification (InSDN)

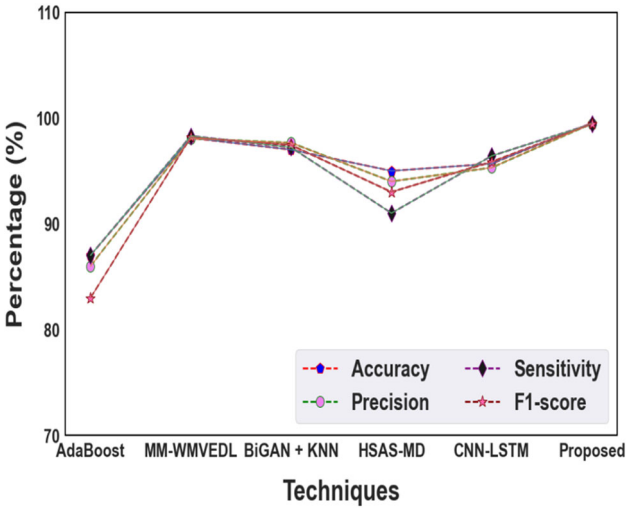


Fig. 7 Evaluation of IoT-23 dataset

Table 5 Comparative Analysis of IoT-23 Dataset

techniques	Accuracy	Precision	Sensitivity	Specificity	F1-score
AdaBoost [35]	87	86	87	–	83
MM-WMVEDL [36]	98.12	98.06	98.31		98.18
BiGAN + KNN [37]	97	97.66	97.22	–	97.44
HSAS-MD [38]	95.00	94.00	91.00	–	93.00
CNN-LSTM [39]	95.68	95.31	96.42	–	95.86
ACNN-LSTM[40]	99	99	99	–	99
DNN[41]	93	95	92	–	93
Proposed	99.53	99.49	99.45	99.48	99.46

models. In Fig. 9, the Roc curves of our suggested model are displayed, clearly demonstrating the relationship between true positives and true negatives. Additionally, it offers a single numerical measurement of the model's total discriminatory ability.

Overall, the findings obtained using the InSDN and IoT-23 datasets show that our technique produces superior outcomes compared to other suggested approaches. Using data augmentation approaches and hyperparameter tuning helped our model produce meaningful results.

4.5 Mitigation process evaluation

In this case, we assessed how well the InSDN and IoT-23 dataset attacks may be mitigated. Figure 10 shows the traffic patterns from the test day where the attack report was generated with the mitigation module turned off and contrasts the patterns with the mitigation turned on.

The green line depicts the traffic created after the mitigation against the attacks, and the red area represents the traffic generated without mitigation policies. The image

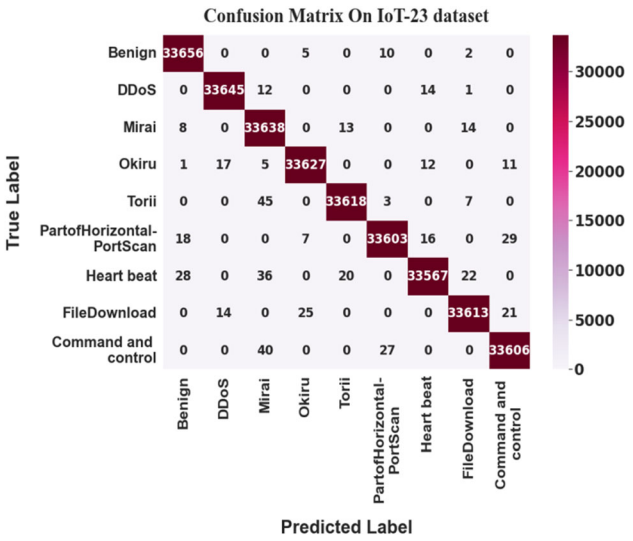


Fig. 8 Confusion Matrix of IoT-23 dataset

makes it easy to comprehend how traffic differed before and after the suggested mitigation method. When the attack happens, network traffic spikes dramatically. The network traffic becomes stable when the mitigation phase is started.

4.6 Discussion

This part discusses the positive features of the recommended strategy versus competing models concerning several performance metrics, including the f1-score, specificity, sensitivity, accuracy, and precision. The trial's outcomes show how effectively the recommended IDS can recognize and mitigate a range of attacks on security in SDN systems. A few key components have been harmoniously combined to produce this outcome. First, the attack is appropriately classified by the EfficientNetV2 and RegNet-based algorithms, demonstrating their effectiveness in extracting pertinent patterns from the provided data. Furthermore, due to the trade-off between model accuracy and processing efficiency, the aforementioned structures are better suited for SDN. Also, the compound scaling approach of EfficientNetV2 allows the system to deal with various data patterns.

Additionally, RegNet allows for customizable learning rates, which enable the model to adapt according to how quickly it absorbs new information from the input. This adaptability is helpful for categorizing intrusions when attack methods and network dynamics change. Additionally, CGAN-based balanced datasets can enhance the framework's generalization performance. Exposure to a wider variety of synthetic cases reduces the likelihood of overfitting and makes the framework more robust and adaptive to different attack strategies. The WHO algorithm's efficient hyperparameter tuning improved the classifier's performance even further. Due to the advantages of these methodologies, our proposed strategy yielded above 99% results for all the performance measures in two different datasets. Specifically, it achieved 99.45% sensitivity for both datasets, showing that the model can detect a substantial portion of true positive events and correctly recognize many actual invasions. Because ignored intrusions can have significant security repercussions, this is crucial for an intrusion detection system to operate well. Furthermore, the success rate of the provided procedure is 99.48 and 99.49% on the InSDN and IoT-23 datasets, respectively, showing that very few regular instances are incorrectly categorized as intrusions. It is an essential feature because it lessens the possibility of unsolicited warnings or activities being triggered in

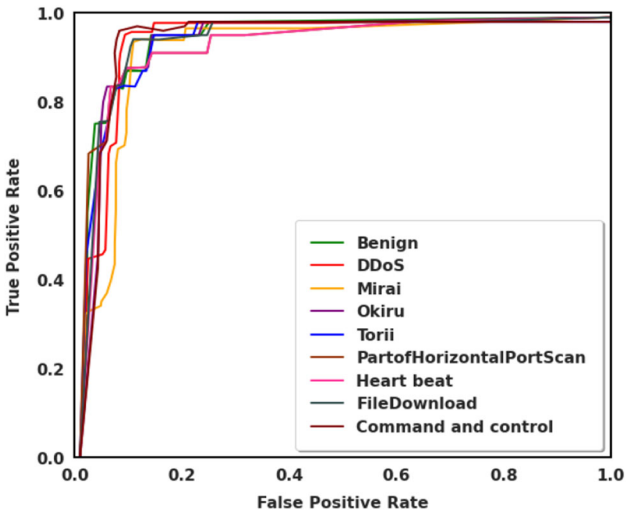


Fig. 9 ROC curve of IoT-23 dataset

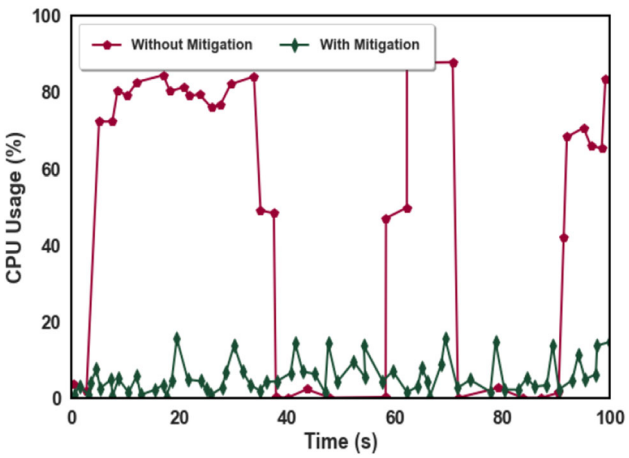


Fig. 10 Mitigation Process Evaluation

response to false alarms. It also reduces the possibility of interfering with regular network activities. Furthermore, ROC plots and confusion matrices produced by the suggested method provide a thorough assessment of the model's functionality. Collectively, these results show that our technique outperforms the benchmarks of earlier methods. Additionally, a congestion management mechanism incorporated into the suggested mitigation strategy can be triggered after the intrusion discovery. Due to this dynamic control, the network's conditions can be adjusted flexibly, reducing bottlenecks and disturbances brought on by malicious activity. This accomplishment validates our technique as a notable advance in the intrusion detection industry and shows how effective our comprehensive procedure is compared to similar studies.

5 Limitations and future directions

Even if our presented framework successfully identifies and classifies various types of intrusions, a few problems still need to be addressed. First, without building any real SDN networks, we used virtual simulation to train and assess the suggested model offline. However, it is crucial to identify online assaults and comprehend how this IDS can respond to an incursion instantly. We plan to test our suggested model in a real-world environment in the future. Furthermore, we use a single SDN controller to verify the framework's performance, but the latency and throughput differ between controllers. Therefore, future research should focus on many controllers to ensure equal understanding and demonstrate how well this approach interacts with other controllers. Furthermore, the proposed approach has only been tested on two datasets. Therefore, our future research will thoroughly assess the model's functionality with other attacks to ensure broader application.

6 Conclusion

SDN-IoT security against attacks can be improved through intrusion detection. This study proposes an intrusion detection model designed explicitly for SDN security utilizing a deep learning-based anomaly detection method that increases IDS accuracy and reduces network intrusions. To validate the efficiency of the suggested IDS, we carried out simulation experiments on the Mininet platform, and the outcomes are contrasted with several existing approaches. Our investigation of comparative performance revealed that the provided model outperformed the other existing models and more clearly identified the several kinds of intrusions in the dataset. For the InSDN and IoT23 datasets, the presented technique achieves an accuracy of 99.56 and 99.53%, respectively. The suggested SDN-based attack mitigation strategy could simultaneously track down and minimize attack sources. It demonstrates how the suggested system efficiently evaluates attack detection precision and speed. Future work on this project will incorporate blockchain technology and more deep learning models to generate an IDS for IoTs that is more effective.

Acknowledgements We declare that this manuscript is original, has not been published before, and is not currently being considered for publication elsewhere.

Author contributions The author confirms sole responsibility for the following: study conception and design, data collection, analysis and interpretation of results, and manuscript preparation.

Appendix

Appendix-I

InSDN dataset					IoT-23 dataset				
Attack types	Original Samples	After removing duplicate record	Newly generated samples	After augmentation	Attack types	Original Samples	After removing duplicate records	Newly generated samples	After augmentation
Normal	68,424	–	0	68,424	Benign	30,858,735	2,113,860	To maintain dataset balance, certain samples are eliminated	68,424
DDoS	121,942	–	To maintain dataset balance, certain samples are eliminated	68,424	DDoS	19,538,713	3,643,225	To maintain dataset balance, certain samples are eliminated	68,424
DoS		–	14,808	68,424	Mirai	9400	9400	59,024	68,424
Probe		–	To maintain dataset balance, certain samples are eliminated	68,424	Okiru	60,990,711	234,942	To maintain dataset balance, certain samples are eliminated	68,424
Botnet	164	–	68,260	68,424	Torii	30	30	68,394	68,424
Password	1405	–	67,019	68,424	PartofHorizontalPortScan	213,853,817	369,525	To maintain dataset balance, certain samples are eliminated	68,424
Web-attack	192	–		68,424	Heart beat	34,518	22,982	45,442	68,424
U2R	17	–	68,407	68,424	FileDownload	71	71	68,353	68,424
					Command and control	21,995	18,939	49,485	68,424

Data availability Not applicable.

Declarations

Conflict of 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.

Ethical approval This material is the author's original work, which has yet to be previously published elsewhere. The paper reflects the author's research and analysis truthfully and completely.

References

1. Luo, K.: A distributed SDN-based intrusion detection system for IoT using optimized forests. PLoS ONE **18**(8), e0290694 (2023)
2. Kumar, C., Biswas, S., Ansari, M.S.A., Govil, M.C.: Nature-inspired intrusion detection system for protecting software-defined networks controller. Comput. Secur. **134**, 103438 (2023)
3. Shaji, N.S., Muthalagu, R., Pawar, P.M.: SD-IIDS: intelligent intrusion detection system for software-defined networks. Multimedia Tools and Applications, 1–33 (2023)
4. Kou, L., Ding, S., Wu, T., Dong, W., Yin, Y.: An intrusion detection model for drone communication network in sdn environment. Drones **6**(11), 342 (2022)
5. Duy, P.T., Khoa, N.H., Do Hoang, H., Pham, V.H.: Investigating on the robustness of flow-based intrusion detection system against adversarial samples using generative adversarial networks. J. Inform. Secur. Appl. **74**, 103472 (2023)
6. Imran, M., Haider, N., Shoaib, M., Razzak, I.: An intelligent and efficient network intrusion detection system using deep learning. Comput. Electr. Eng. **99**, 107764 (2022)
7. Talukder, M.A., Hasan, K.F., Islam, M.M., Uddin, M.A., Akhter, A., Yousuf, M.A., Moni, M.A.: A dependable hybrid machine learning model for network intrusion detection. J. Inform. Security Appl. **72**, 103405 (2023)
8. Bhardwaj, A., Tyagi, R., Sharma, N., Khare, A., Punia, M.S., Garg, V.K.: Network intrusion detection in software defined networking with self-organized constraint-based intelligent learning framework. Meas. Sens **24**, 100580 (2022)
9. Kasongo, S.M.: A deep learning technique for intrusion detection system using a recurrent neural networks based framework. Comput. Commun. **199**, 113–125 (2023)
10. Alzahrani, A.O., Alenazi, M.J.: ML-IDSDN: Machine learning based intrusion detection system for software-defined network. Concurr. Comput. Pract. Exp. **35**(1), e7438 (2023)
11 Tayfour, O.E., Mubarakali, A., Tayfour, A.E., Marsono, M.N., Hassan, E., Abdelrahman, A.M.: Adapting deep learning-LSTM method using optimized dataset in SDN controller for secure IoT. Soft Comput. (2023). https://doi.org/10.1007/s00500-023-08348-w
12. Bour, H., Abolhasan, M., Jafarizadeh, S., Lipman, J., Makhdoom, I.: A multilayered intrusion detection system for software defined networking. Comput. Electr. Eng. **101**, 108042 (2022)
13. Qureshi, S.S., He, J., Qureshi, S., Zhu, N., Zardari, Z.A., Mahmood, T., Wajahat, A.: SDN-enabled deep learning based detection mechanism (DDM) to tackle DDoS attacks in IoTs. J. Intell. Fuzzy Syst. **44**(6), 10675–10687 (2023)
14. Alshammari, T.M., Alserhani, F.M.: Scalable and robust intrusion detection system to secure the iot environments using software defined networks (SDN) enabled architecture. Int. J. Comput. Networks Appl **9**(6), 678–688 (2022)
15. Jadhav, K.P., Arjariya, T., Gangwar, M.: Hybrid-Ids: an approach for intrusion detection system with hybrid feature extraction technique using supervised machine learning. Int. J. Intell. Syst. Appl. Eng. **11**(5s), 591–597 (2023)
16. Wang, J., Wang, L.: SDN-Defend: a lightweight online attack detection and mitigation system for DDoS attacks in SDN. Sensors **22**(21), 8287 (2022)
17. Al Razib, M., Javeed, D., Khan, M.T., Alkanhel, R., Muthanna, M.S.A.: Cyber threats detection in smart environments using SDN-enabled DNN-LSTM hybrid framework. IEEE Access **10**, 53015–53026 (2022)
18. Hnamte, V., Hussain, J.: Dependable intrusion detection system using deep convolutional neural network: a Novel framework and performance evaluation approach. Telemat. Inform Rep. **11**, 100077 (2023)
19. Maheshwari, A., Mehraj, B., Khan, M.S., Idrisi, M.S.: An optimized weighted voting based ensemble model for DDoS attack detection and mitigation in SDN environment. Microprocess. Microsyst. **89**, 104412 (2022)
20. Ravi, V., Chaganti, R., Alazab, M.: Deep learning feature fusion approach for an intrusion detection system in SDN-based IoT networks. IEEE Internet Things Mag. **5**(2), 24–29 (2022)
21. Maray, M., MesferAlshahrani, H., Alissa, A.K., Alotaibi, N., Gaddah, A., Meree, A., Ahmed Hamza, M.: Optimal deep learning driven intrusion detection in SDN-Enabled IoT environment. Comput. Mater. Continua. **74**(3), 6587–6604 (2022)
22. Logeswari, G., Bose, S., Anitha, T.: An intrusion detection system for sdn using machine learning. Intell. Autom. Soft Comput. **35**(1), 867–880 (2023)
23. Aslam, M., Ye, D., Tariq, A., Asad, M., Hanif, M., Ndzi, D., Jilani, S.F.: Adaptive machine learning based distributed denial-of-services attacks detection and mitigation system for SDN-enabled IoT. Sensors **22**(7), 2697 (2022)
24. El Sayed, M.S., Le-Khac, N.A., Azer, M.A., Jurcut, A.D.: A flow-based anomaly detection approach with feature selection method against ddos attacks in sdns. IEEE Trans. Cognitive Commun. Netw. **8**(4), 1862–1880 (2022)
25. Elsayed, R.A., Hamada, R.A., Abdalla, M.I., Elsaid, S.A.: Securing IoT and SDN systems using deep-learning based automatic intrusion detection. Ain Shams Eng. J. **14**(10), 102211 (2023)
26 Ferrão, T., Manene, F., Ajibesin, A.A.: Multi-attack intrusion detection system for software-defined internet of things network. Comput. Mater. Continua. (2023). https://doi.org/10.32604/cmc.2023.038276
27. Arun Prasad, P.B., Mohan, V., Vinoth Kumar, K.: Hybrid metaheuristics with deep learning enabled cyberattack prevention in software defined networks. Tehnički Vjesnik **31**(1), 208–214 (2024)
28. Polat, O., Türkoğlu, M., Polat, H., Oyucu, S., Üzen, H., Yardımcı, F., Aksöz, A.: Multi-stage learning framework using convolutional neural network and decision tree-based classification for detection of DDoS pandemic attacks in SDN-based SCADA systems. Sensors **24**(3), 1040 (2024)
29. Said, R.B., Sabir, Z., Askerzade, I.: CNN-BiLSTM: A hybrid deep learning approach for network intrusion detection system in software defined networking with hybrid feature selection. IEEE Access. **11**, 138732–138747 (2023). https://doi.org/10.1109/ACCESS.2023.3340142
30. Friha, O., Ferrag, M.A., Shu, L., Maglaras, L., Choo, K.K.R., Nafaa, M.: FELIDS: Federated learning-based intrusion detection system for agricultural internet of things. J. Parallel Distrib. Comput. **165**, 17–31 (2022)
31. Abdallah, M., An Le Khac, N., Jahromi, H., Delia Jurcut, A.: A hybrid CNN-LSTM based approach for anomaly detection

systems in SDNs. In: Proceedings of the 16th International Conference on Availability, Reliability and Security, p 1–7 (2021)
32. Safwan, H., Iqbal, Z., Amin, R., Khan, M.A., Alhaisoni, M., Alqahtani, A., Chang, B.: An IoT environment based framework for intelligent intrusion detection. CMC Comput. Mater. Continua. **75**(2), 2365–2381 (2023)
33. Alshra'a, A.S., Farhat, A., Seitz, J.: Deep learning algorithms for detecting denial of service attacks in software-defined networks. Procedia Comput. Sci. **191**, 254–263 (2021)
34. Elsayed, M.S., Jahromi, H.Z., Nazir, M.M., Jurcut, A.D.: The role of CNN for intrusion detection systems: An improved CNN learning approach for SDNs. In: International Conference on Future Access Enablers of Ubiquitous and Intelligent Infrastructures. Springer International Publishing, Cham, p 91–104 (2021)
35. Gyamfi, E., Jurcut, A.D.: Novel online network intrusion detection system for industrial iot based on oi-svdd and as-elm. IEEE Internet Things J. **10**(5), 3827–3839 (2022)
36. Sanju, P.: Enhancing intrusion detection in IoT systems: a hybrid metaheuristics-deep learning approach with ensemble of recurrent neural networks. J. Eng. Res. (2023). https://doi.org/10.1016/j.jer.2023.100122
37. Abdalgawad, N., Sajun, A., Kaddoura, Y., Zualkernan, I.A., Aloul, F.: Generative deep learning to detect cyberattacks for the IoT-23 dataset. IEEE Access **10**, 6430–6441 (2021)
38. Hamza, A.A., Abdel Halim, I.T., Sobh, M.A., Bahaa-Eldin, A.M.: HSAS-MD analyzer: a hybrid security analysis system using model-checking technique and deep learning for malware detection in IoT apps. Sensors **22**(3), 1079 (2022)
39. Sahu, A.K., Sharma, S., Tanveer, M., Raja, R.: Internet of things attack detection using hybrid deep learning model. Comput. Commun. **176**, 146–154 (2021)
40. Kolhar, M., Aldossary, S.M.: DL-Powered anomaly identification system for enhanced IoT data security. Comput. Mater. Continua. **77**(3), 2857–2857 (2023)
41. Bhandari, G., Lyth, A., Shalaginov, A., Grønli, T.M.: Distributed deep neural-network-based middleware for cyber-attacks detection in smart IoT ecosystem: a novel framework and performance evaluation approach. Electronics **12**(2), 298 (2023)

Baswaraju Swathi graduated in B.E Computer Science and Information Technology in 2005 from JNTU Hyderabad, Pursued her M.Tech in Software Engineering in JNTU Hyderabad. She was awarded with Ph.D. from Jain University in 2023 in the area of Soft Computing Techniques. She is a VMware Vsphere Technologies certified trainer. She has NPTEL Certifications in Java Programming, Industrial Internet of Things, Cloud Computing, Software Testing and many more. She has academic and research experience of over 13 years. Currently she is heading Department of CSE (Data Science) in New Horizon College of Engineering. Her research interest includes Soft Computing, Software Engineering, Machine Learning, Data Science, has published numerous papers in international journals.

Soma Sekhar Kolisetty working as Assistant Professor, Department of Computer Science and Engineering, University College of Engineering Narasaraopet, JNTUK, India. He received his M.Tech. in Software Engineering and B.Tech. in Computer Science and Engineering from Jawaharlal Nehru Technological University, Hyderabad, in 2011 and 2009. He had an academic, industrial and research experience of 8 years. His research interests include Distributed systems, Parallel computing, Machine learning, and data science.

G Venkata Sivanarayana is currently working as an Assistant Professor in the Department of Computer Science and Engineering since 2011, School of Technology, GITAM, Deemed to be University, Visakhapatnam, India. He received M.Tech degree from Andhra University. He is Pursuing Ph.D in GITAM (Deemed to be University). The author has contributed a good number of research papers indexed in Scopus, Springer Book Series chapter, etc. and member of various Professional Societies. He also published an Australian Patent. He also presented papers at various conferences. His research interests include Image Analysis, Machine Learning and Deep Learning.

Srinivasa Rao Battula working as Professor, School of computer and Information Sciences (SCIS), University of Hyderabad, Gachibowly, Hyderabad-500046, India. His research interests are Soft Computing, Image Processing, Machine learning and Deep learning.