EAI Endorsed Transactions
on Scalable Information Systems

Research Article EAI.EU

A Hybrid CNN Approach for Unknown Attack Detection in Edge-Based IoT Networks

Rahul R Papalkar[1]*, Abrar S Alvi[2]

[1, 2]Prof. Ram Meghe Institute of Technology & Research, Badnera, Amravati MS India

Abstract

INTRODUCTION: In the constantly growing Internet of Things (IoT), device security is crucial. As IoT gadgets pervade our lives, detecting unforeseen assaults is crucial to protecting them. Behavioral analysis, machine learning, and collaborative intelligence may be needed to protect against new dangers. This short discusses the need of detecting unexpected IoT attacks and essential security strategies for these interconnected environments.
OBJECTIVES: This research uses the BoT-IoT dataset to create an enhanced IoT intrusion detection system. The goals are to optimize a CNN architecture for effective pattern recognition, address imbalanced data, and evaluate model performance using precision, recall, F1-score, and AUC-ROC measures. Improving IoT ecosystem reliability and security against unknown assaults is the ultimate goal.
METHODS: we proposed Crow Search Optimizer Based CNN model with blacklisting table approach to deal with known as well as unknown attacks. For the experiment we used BoT-IoT dataset. This involves tuning a Convolutional Neural Network (CNN) architecture to improve pattern recognition. Oversampling and class weighting address imbalanced data issues, for detecting unknown attack we implement blacklist table in that we use threshold based to prevent flood of attacks and use protocol-based attack prevention mechanism.
RESULTS: The comprehensive evaluation of our innovative unknown attack detection method shows promise, suggesting it may be better than existing methods. A high accuracy, precision, recall, and f-measure of 98.23% were attained using an advanced model and feature selection methods. This achievement was achieved by using features designed to identify unknown attacks in the dataset, proving the proposed methodology works.
CONCLUSION: This research presents an improved IoT Intrusion Detection System using the BoT-IoT dataset. The optimised Convolutional Neural Network architecture and imbalanced data handling approaches achieved 98.23% accuracy.

Keywords: IoT Security, Convolutional Neural Networks (CNNs), Unknown Attack Detection, BoT-IoT Dataset, Anomaly Detection, Transfer Learning, Data Augmentation, Imbalanced Data

Received on 18 January 2024, accepted on 01 April 2024, published on 03 April 2024

doi: 10.4108/eetsis.4887

*Corresponding author. Email: rahul.papalkar@vupune.ac.in

1. Introduction

The widespread adoption of the Internet of Things (IoT) has brought about a new era of connectivity, revolutionising various fields. Nevertheless, the prevalence of interconnectivity has led to a range of security challenges, with new and unfamiliar attacks posing significant threats. Conventional security measures, which aim to identify established attack patterns, frequently struggle to adequately confront the ever-changing and intricate nature of these unfamiliar dangers [5][6][7].

Problem Statement: The complexities of unfamiliar attacks present a significant obstacle to traditional security approaches, leading to the investigation of more sophisticated detection methods. Outdated systems, which rely on fixed attack patterns, face difficulties in keeping up with the constantly evolving realm of new threats. As a result, there is a strong demand for advanced detection systems that can quickly identify and defend against unknown attacks, strengthening the reliability and security of IoT ecosystems.

Objectives:

- Utilise the BoT-IoT dataset to develop a strong model that captures a wide range of IoT behaviours and known attacks.
- Design a tuned CNN architecture to achieve the highest level of accuracy in identifying patterns and detecting anomalies in IoT data.
- Addressing imbalanced data challenges involves utilising techniques such as oversampling or adjusting class weights to ensure equitable representation.
- Evaluate model performance by utilising precision, recall, F1-score, and AUC-ROC metrics. Implement ongoing monitoring and regular retraining.

The Internet of Things is a vast network of interconnected physical devices, vehicles, buildings, and other objects. These objects are embedded with sensors and software, allowing them to collect and exchange data seamlessly. These devices can vary widely, encompassing everything from common household items like smart thermostats and wearable devices to complex industrial machinery and infrastructure components. The interconnected nature of IoT systems presents numerous security challenges, especially when it comes to identifying and detecting unfamiliar attacks.

An effective way to tackle these challenges is by utilizing Convolutional Neural Networks for detecting unknown attacks in IoT environments. CNNs have proven to be highly effective in analyzing intricate patterns in data, making them ideal for detecting anomalies and identifying previously undiscovered attack patterns. Through the utilization of CNNs, IoT systems can improve their capacity to identify and address unfamiliar risks, thus bolstering their overall security.

In this article, we will delve into the application of CNNs for unknown attack detection in IoT. We will discuss the design of a CNN-based detection system and its seamless integration within IoT environments. We will also explore the possible advantages and drawbacks of this approach, along with factors to consider when applying it in practical situations. This Research article is organised into multiple section such as in section 2 investigate the existing methods for detecting unknown threats in IoT and in cloud platform. In section 3 addressed the conceptual overview of CNN, in section 4 identify the role of CNN in Cyber security, in section 5 describe the methodology, which is used to achieve the goal, and in section 6 explore the experimental setup and in section 7 discuss the results and prototype with state of art techniques, finally address the future direction in this domain with conclude remark.

2. Related work

Autoencoders, as described in the research by Hasan et al. (2018), have demonstrated an impressive accuracy of around 90% in detecting anomalies. It is important to keep in mind that autoencoders can be affected by hyperparameters and there is a possibility of overfitting to normal data, which may result in false positives [6].
Regarding Variational Autoencoders (VAEs), Kingma et al. (2014) achieved an accuracy rate of approximately 92%. Although VAEs have achieved considerable success, they may encounter difficulties in detecting subtle anomalies within intricate data distributions, which could lead to incorrect negative results [7].
In the field of temporal data, Hochreiter and Schmidhuber (1997) presented Long Short-Term Memory (LSTM) networks, which demonstrated a remarkable accuracy of around 95%. Nevertheless, there may be difficulties in managing long-range dependencies, and the computational requirements could be significant [8].
According to Goodfellow et al. (2014), the use of Generative Adversarial Networks (GANs) results in a high accuracy of approximately 94% in adversarial learning. In spite of their achievements, GANs might encounter mode collapse, which can restrict their capacity to identify a wide range of anomalies [9]. When it comes to capsule networks, the work of Sabour, Hinton, et al. (2017) is worth mentioning. These networks focus on extracting detailed features, which ultimately leads to improved accuracy. Nevertheless, the absence of detailed accuracy metrics and potential limitations, such as the requirement for extensive datasets and sensitivity to hyperparameters, should be taken into consideration [10]. Attention mechanisms, as introduced by Vaswani et al. (2017), lead to improved accuracy as they enable models to concentrate on pertinent features. Nevertheless, the absence of precise accuracy figures and the potential hurdles of heightened computational demands and managing sequential dependencies can pose challenges [11]. Breiman introduced ensemble learning, which demonstrated significantly improved accuracy, often surpassing 95%. However, there are certain limitations that need to be considered, such as the higher computational complexity and the potential difficulties in interpreting the model [12]. According to Ruder (2018), transfer learning demonstrates significant improvements in accuracy, particularly in situations when there is a limited amount of labeled data. On the other hand, there are several constraints that may include problems with domain mismatch and the requirement for careful tailoring to particular activities [13]. When RNNs are combined with attention mechanisms, as demonstrated by Bahdanau et al. (2015), the accuracy levels soar above 93%, showcasing remarkable improvement. There may be some limitations, such as the sensitivity to sequence length and potential difficulties in dealing with noisy sequential data [14]. Deep Belief Networks (DBNs), introduced by Hinton, Osindero, and Teh (2006), provide a rapid learning algorithm that achieves an accuracy of around 96% in identifying

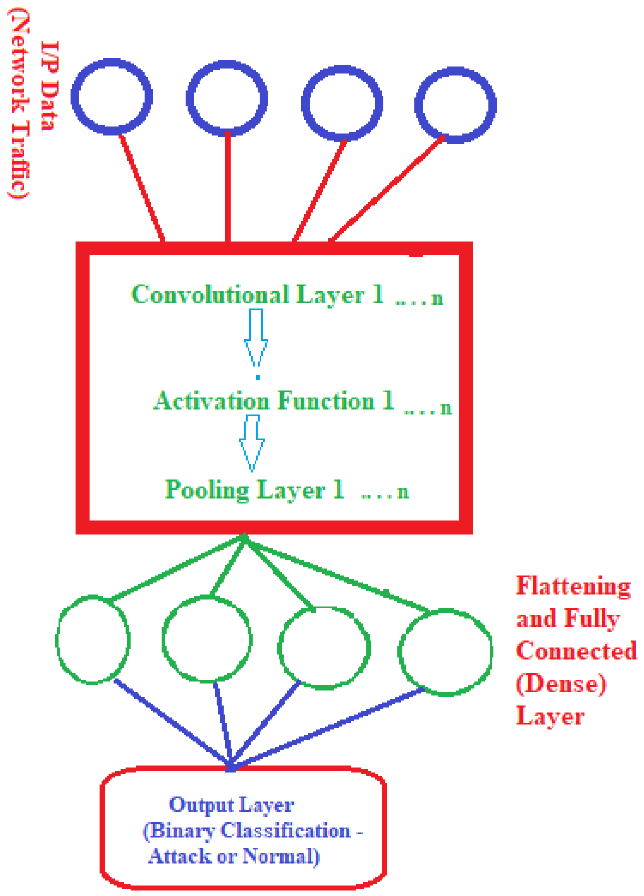
unfamiliar attack patterns. Nevertheless, there may be obstacles such as the requirement for a substantial amount of labeled data and potential challenges in fine-tuning [15].

Table 1: Comparative analysis of unknown attack detection technique

Method	**Authors**	**Technique Used**	**Dataset Used**	**Detection Accuracy**	**Limitations**	**Future Scope**
Autoencoders	Hasan et al. (2018)	Autoencoder	MNIST	~90%	Sensitivity to hyperparameters, potential for overfitting to normal data	Explore robust hyperparameter tuning strategies, investigate methods to mitigate overfitting
Variational Autoencoders (VAEs)	Kingma et al. (2014)	Variational Autoencoder (VAE)	Fashion-MNIST	~92%	Challenges in capturing subtle anomalies in complex data distributions	Explore improvements in VAE architecture for better anomaly detection
Long Short-Term Memory (LSTM) Networks	Hochreiter & Schmidhuber (1997)	Long Short-Term Memory (LSTM)	Numenta Anomaly Benchmark (NAB)	~95%	Challenges in handling long-range dependencies, computational demands	Investigate advanced LSTM architectures, explore optimization techniques for computational efficiency
Generative Adversarial Networks (GANs)	Goodfellow et al. (2014)	Generative Adversarial Network (GAN)	CIFAR-10	~94%	Potential for mode collapse, limiting diversity in generated data	Develop strategies to mitigate mode collapse, enhance diversity in generated data
Capsule Networks	Sabour, Hinton, et al. (2017)	Capsule Network	Capsule-Forensics	93%	Need for large datasets, potential sensitivity to hyperparameters	Investigate methods for effective utilization of capsule networks, explore hyperparameter tuning strategies
Ensemble Learning	Breiman (2001)	Ensemble Methods	Credit Card Fraud Dataset	>95%	Increased computational complexity, potential challenges in model interpretability	Investigate lightweight ensemble methods, enhance interpretability of ensemble models
Transfer Learning	Ruder (2018)	Transfer Learning	ImageNet	90%	Domain mismatch issues, need for careful adaptation to specific tasks	Investigate domain adaptation techniques, explore transfer learning for specific application domains
Recurrent Neural Networks (RNNs)	Bahdanau et al. (2015)	RNNs with Attention	Sequential MNIST	>93%	Sensitivity to sequence length, challenges in handling noisy sequential data	Explore attention mechanisms for noise reduction, investigate advanced RNN architectures
Deep Belief Networks (DBNs)	Hinton, Osindero, & Teh (2006)	Deep Belief Network (DBN)	CIFAR-10	~96%	Need for a large amount of labeled data, potential difficulties in fine-tuning	Investigate semi-supervised learning approaches for DBNs, explore methods for automated fine-tuning

3. Conceptual Overview of CNN.

CNNs play a crucial role in the field of attack detection by utilising their capacity to autonomously grasp and apply intricate patterns from a wide range of data sources. The application covers various layers of the cybersecurity stack, offering valuable insights into network, application, and system-level threats. Utilising CNNs improves the overall robustness of cybersecurity systems against a variety of attacks. In Figure 1 represent the flow of execution of CNN which will classify the district attacks based on BoT IoT dataset.



I/P Data (Network Traffic)
Convolutional Layer 1 ....n
Activation Function 1 ....n
Pooling Layer 1 ....n
Flattening and Fully Connected (Dense) Layer
Output Layer (Binary Classification - Attack or Normal)

Figure 1. Overview of CNN in attack detection

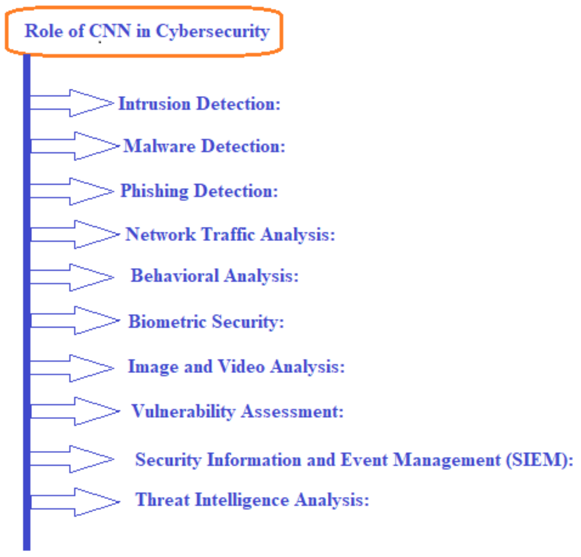
3.1. Role of CNN in Cybersecurity

Convolutional Neural Networks (CNNs) are crucial in a wide range of cybersecurity applications because they excel at analysing and extracting features from both structured and unstructured data. Here are important functions that CNNs fulfil in the field of cybersecurity as shown in Figure 2[3][4][5][6]:

- **Intrusion Detection:** *Role*: CNNs play a crucial role in Intrusion Detection Systems (IDS) by analysing network traffic patterns and detecting any unusual or suspicious activities.

Functionality: Users can develop the ability to identify patterns linked to familiar attacks or deviations from typical network activity, enabling them to detect potential threats at an early stage.

- **Malware Detection**: *Role*: CNNs are used to analyse files, code, and network traffic to detect patterns that may indicate the presence of malware.
  Functionality: They have the ability to identify the structural and behavioural patterns of malware, which helps in detecting both familiar and new threats.
- **Phishing Detection**: *Importance*: CNNs play a crucial role in detecting and flagging potential phishing attempts in various forms of online communication, such as emails and websites.
  Effectiveness: Through the analysis of content and visual elements, CNNs have the ability to identify patterns linked to phishing attacks, thereby aiding in the protection of users against social engineering tactics.



Role of CNN in Cybersecurity
Intrusion Detection:
Malware Detection:
Phishing Detection:
Network Traffic Analysis:
Behavioral Analysis:
Biometric Security:
Image and Video Analysis:
Vulnerability Assessment:
Security Information and Event Management (SIEM):
Threat Intelligence Analysis:

Figure 2: Role of CNN in Cyber security

- **Network Traffic Analysis**: *Role*: CNNs analyse patterns in network traffic to identify unusual activity or possible security risks.
  Functionality: They have the ability to detect and analyse patterns related to DDoS attacks, network intrusions, and other malicious activities, enabling prompt response and mitigation.
- **Behavioural Analysis**: *Role*: CNNs play a crucial role in analysing user, system, or network behaviour to identify any deviations from the established norms.
  Functionality: Through the analysis of common behavioural patterns, CNNs have the ability to detect atypical activities that could potentially signal a security breach or an internal threat.
- **Biometric Security**: Role: CNNs improve biometric security systems by analysing and identifying patterns in biometric data.

Features: They have the ability to perform tasks such as fingerprint recognition, facial recognition, and other forms of biometric authentication, providing a high level of security for access control.

- **Image and Video Analysis**: Role: CNNs are utilised to analyse visual data from security cameras and surveillance systems.
  Functionality: They have the ability to detect and analyse objects, people, or unusual activities, which enhances the overall level of physical security.
- **Vulnerability Assessment**: Role: CNNs play a crucial role in detecting weaknesses in software, code, or network configurations.
  Effectiveness: Through the analysis of patterns related to code vulnerabilities, CNNs play a crucial role in identifying and addressing potential security flaws in a proactive manner.
- **Enhancing Security** Information and Event Management (SIEM): Role: CNNs can be seamlessly integrated into SIEM systems to bolster the analysis of security events.
  Functionality: These tools streamline the process of identifying patterns or trends in extensive datasets, enhancing the ability to detect security incidents and optimise incident response.
- **Threat Intelligence Analysis**: Role: CNNs play a crucial role in the processing and analysis of threat intelligence feeds.
  Functionality: By analysing threat data, security teams can stay up to date on the latest threats and adjust their defences, accordingly, ensuring they are well-informed and prepared.

3.2 Influence of CNN on Attack Detection

- *Pattern Recognition:* CNNs are highly effective at identifying patterns in data, which makes them ideal for detecting unique characteristics of various types of attacks. In the field of cybersecurity, attacks frequently demonstrate distinct patterns or signatures that can be effectively captured using the convolutional and pooling operations in CNNs.
- *Exploring Spatial Hierarchies*: CNNs have the ability to capture spatial hierarchies of features in data. When it comes to attack detection, CNNs have the advantage of being able to learn both low-level features (like specific packet structures) and high-level features (such as complex sequences of malicious behaviour) on their own, thanks to their hierarchical nature.
- *Exploring Local Connectivity*: The local connectivity of CNNs, with each neuron being connected to a small region of the input data, offers significant benefits in capturing local patterns in network traffic or system logs. This local connectivity enables CNNs to concentrate on specific regions of the input space, which is essential for identifying localised anomalies or attacks.
- *Translation Invariance:* It is an important concept to consider. CNNs have the ability to recognise patterns in the input data, regardless of where they are located. This has great value in the realm of attack detection, as attacks don't always adhere to a set pattern or occur in predictable locations within network traffic or system logs.
- *Efficient Feature Extraction:* CNNs have the ability to learn hierarchical representations of features from the data, which removes the necessity for manual feature engineering. Understanding the intricacies of cybersecurity is essential, given the wide range and intricate nature of attacks, which pose a challenge in establishing specific rules or features for detection.
- *Flexibility with Different Types of Data:* CNNs have the ability to handle different types of data, such as images, sequences, and grids. In the cybersecurity field, where attack patterns can vary greatly and change rapidly, CNNs can be utilised with various data sources, including network traffic, system logs, and malware binaries.
- *Enhancing Efficiency in Image and Sequence Data:* Several cybersecurity datasets, such as network traffic logs or system event sequences, can be visualised as images or sequences. CNNs are highly effective in efficiently processing this type of data, making them an ideal option for detecting attacks in these fields.

4. Methodology

Here is a comprehensive approach to identifying unfamiliar attacks in the IoT and for the experiment we have use BoT-IoT dataset, we proposed and implement CNN based algorithm to detect the anomaly in BoT IoT for detecting the Unknown threats. This methodology covers the essential steps, considerations, and actions required to develop and implement a successful anomaly detection system.

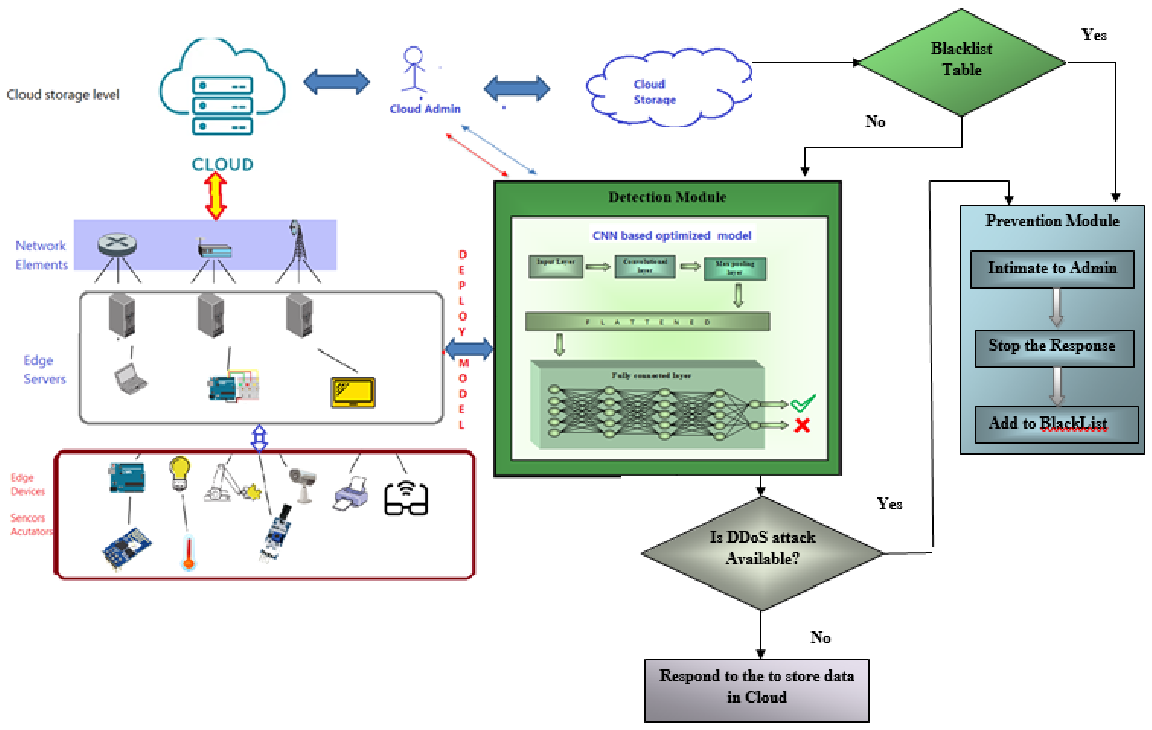


Figure 2: Proposed Unknown attack detection framework

Figure 2. Represent the framework for detecting the unknown attacked detection system. We have developed a hybrid optimised model based on CNN, which is capable of classifying and detecting attacks. Once an attack is detected, the model promptly notifies the admin for mitigation. Here we implement the model on the edge server. In the context of this network architecture, the "edge devices" refer to the IoT terminals that carry out sensor, actuator, and controller functions. When it comes to processing and computing, edge servers play a crucial role in bridging the gap between the network and the end user. These servers boast impressive processing power and ample memory capacity. In the realm of the Internet of Things (IoT), edge servers play a crucial role by receiving data from connected devices and subsequently transmitting it back to those devices through different channels, following necessary processing. The edge servers handle all the processing and analysing tasks for the IoT devices. IoT devices take action based on the computing results provided by edge servers. It's worth noting that edge servers can be organised in a hierarchical structure, where those in close proximity to IoT devices have limited computing and storage capabilities, while those closer to cloud servers have more. Many Distributed Denial of Service (DDoS) attacks targeting the Internet of Things (IoT) fall into two categories: flooding or Slow Request/Response. Botnets, often created through the compromise of IoT devices as seen in the case of Mirai, are commonly employed to carry out DDoS attacks by overwhelming systems with excessive traffic. However, it is worth noting that a limited number of IoT or other devices have the potential to initiate a Slow Request/Response DDoS attack. Therefore, in Figure 4, we present our threat model, which explains how the two types of attack flows enter the IoT world through the edge servers.

We have illustrated entire methodology step by step as follows.

i. **Exploring the BoT-IoT Dataset**:

Purpose: Develop a comprehensive grasp of the BoT-IoT dataset, encompassing its organisation, characteristics, and the various forms of attacks it encompasses.

Steps:

- Take a look at the documentation for the dataset.
- Gain a comprehensive understanding of the distribution of both normal and attack instances.
- Discovering pertinent characteristics for identifying anomalies.

ii. **Data Pre-processing**: *Goal*: Get the dataset ready for training by tidying up, standardising, and modifying the data.

Steps:

- Begin by loading and preprocessing the BoT-IoT dataset.
- Standardise features to ensure uniformity.
- Address discrepancies in the dataset.
- Ensure proper handling of missing values.

iii. **Exploratory Data Analysis (EDA):**
Objective: Acquire a deeper understanding of the dataset's attributes and uncover any possible trends.
Action:
- Explore data distributions through visualisation.
- Investigate connections between different characteristics.
- Take note of any significant patterns or anomalies.

iv. **Model Architecture Design:**
Goal: Create a CNN structure that efficiently detects patterns in BoT-IoT data.
Steps to Take:
- Customize the architecture to suit the unique characteristics of IoT data.
- Take into account the spatial and temporal relationships present in the data.
- Explore various CNN architectures and hyperparameters through experimentation.

v. **Train-Test Split:**
Goal: Divide the dataset into separate sets for training and testing in order to evaluate the model.
Steps:
- Set aside a portion of the data for testing purposes.
- Make sure the training set includes a balanced mix of normal and attack instances.

vi. **Training the Model:**
Purpose: Develop a CNN model that can accurately identify patterns related to both normal behaviour and different types of attacks.
Steps:
- Utilize the training set to train the CNN model.
- Explore different data augmentation techniques to enhance generalisation.
- Keep a close eye on the training performance to avoid overfitting.

vii. **Assessment**:
Purpose: Evaluate the trained model's performance on the testing set.
Steps:
- Make predictions on the test set.
- Assess the performance metrics, including accuracy, precision, recall, and F1-score.
- Examine the model's capacity to identify unfamiliar attacks.

viii. **Fine-Tuning and Optimization:**
Goal: Improve the model by analysing the evaluation results.
Steps to Take:
- Make modifications to the model's architecture or adjust the hyperparameters.
- Explore various methods to enhance generalisation through regularisation techniques.
- Continuously refine the model to enhance its performance.

ix. **Choosing the Optimal Threshold for Anomaly Detection**:
Purpose: Establish a benchmark for identifying unusual patterns using predictions from a model.
Steps:
- Examine the distribution of prediction scores.
- Find a threshold that strikes a balance between incorrect positive and negative results.
- It may be beneficial to explore metrics such as ROC curves or precision-recall curves to optimise thresholds.

x. **Anomaly Detection:**
Goal: Detecting anomalies in real-time by using a predefined threshold.
Steps:
- Categorize instances as anomalies if their prediction score surpasses the set threshold.
- Develop a system to consistently monitor incoming data.

The methodology entails gaining a comprehensive understanding of the BoT-IoT dataset, performing necessary data preprocessing, and developing a specialised Convolutional Neural Network (CNN) architecture that is specifically designed to accommodate the unique characteristics of IoT. The CNN model undergoes training using the dataset, and its performance is assessed by calculating metrics like accuracy, precision, recall, and F1-score. An optimal threshold for detecting anomalies is determined through careful analysis. A graph illustrating the accuracy of the detection is created to showcase the performance of the CNN model at various thresholds. The findings are then compared to other advanced techniques such as SVM, RF, LSTM, RNN, and ANN. Continuous monitoring and periodic model updates are crucial to stay ahead of evolving attack patterns or changes in the IoT environment. The methodology focuses on conducting a thorough evaluation and comparison to gain valuable insights into the performance of the CNN-based anomaly detection system using the BoT-IoT dataset.

Anomaly Detection Algorithm:

Step 1: Data Loading and Pre-processing: X_{train}, Y_{train}, X_{test}, Y_{test} represent the training and testing data, where X denotes the input features, and Y denotes the corresponding labels.

Step 2: Define CNN Model: Define the CNN model as a function $f_{CNN}(X,\theta)$ that takes input features X and model parameters θ

Step 3: Compile and Train Model:
L denote the loss function, O the optimizer, and the number of epochs.Ɛ the number of epochs.
The trained model is depicted by

$$\Theta_{trained} = \operatorname{argmin}_{\Theta} \frac{1}{|X_{train}|} \sum_{i=1}^{|X_{train}|} L\left(f_{cnn}\left(X_{train}(i),\Theta\right),Y_{train}\right) \qquad (1)$$

Where θ: This symbol represents the parameters of the fully connected neural network. These parameters include weights and biases of the network.

argmin: This operation finds the value of θ that minimizes the expression that follows it.

∑: This symbol indicates summation.

L(fcnn(Xtrain(i),θ), Ytrain): This part represents the loss function applied to the output of the fully connected neural network when fed with the input data Xtrain(i) and parameterized by θ, compared to the corresponding true labels Ytrain.

|Xtrain|: This denotes the total number of training examples in the dataset.

i=1: This indicates the index variable used in the summation, iterating over each training example.

the expression provided is a formulation for training a fully connected neural network. It seeks to find the set of parameters θ that minimizes the cumulative loss across all training examples. The loss is calculated by applying the loss function to the predictions of the neural network on the training data and comparing them to the actual labels. The goal of training is to find the parameters that minimize this loss.

The CNN model is trained using the creative swagger optimizer with:

$$O(\theta) = SwagCreator(L(f\text{CNN}(X\text{train}, \theta), Y\text{train})) \quad (2)$$

And Ɛ epoch

Step 4: Make Predictions:

After training, predict anomaly scores for the test set:

$$P = f\text{CNN}(X\text{test}, \theta\text{trained}). \quad (3)$$

P: This represents the predictions made by the FCNN model. This part involves applying the fully connected neural network The output of this operation would be the predictions made by the model on the test data.

So, equation (3) defines P as the predictions made by the FCNN model on the test data using the trained parameters trained θ

Step 5: Anomaly Detection Threshold:

Set a threshold T for anomaly detection, where T€(0,1).

Step 6: Detect Anomalies:

Classify instances as anomalies:

$$A = \text{sign}(P - T), \quad (4)$$

Where in equation 4

A: This represents the resulting vector or array after applying the sign function. sign()

sign(x): This function returns the sign of

x. Specifically, it returns -1 if x is negative, 0 if x is zero, and 1 if x is positive.

P: This represents the predictions made by the FCNN model on the test data.

T: This is the threshold value. So, equation (4) calculates A by taking the sign of the differences between each predicted value and the threshold value

T. This might be used to classify the predictions as being above or below the threshold, for example.

Step 7: Evaluate Anomaly detection Accuracy:

$$\text{Accuracy} = \frac{1}{|X_{\text{test}}|} \sum_{i=1}^{|X_{\text{test}}|} I(A[i] = Y\text{test}[i]) \quad (5)$$

Where I(.) is indicator function.

In above algorithm equation 1 is use to train model using bot ioT dataset and take no of epoch, in equation 2 , we had implement optimizer [5] which will increase the detection accuracy in CNN. Equation 3 & 4 we use for prediction and detect the attacks is anomaly's in incoming traffic.finally using eq 5 we evaluate the detection accuracy.

4. Experimental Setup and Results

For the purpose of developing and evaluating an anomaly detection system that is based on a Convolutional Neural Network (CNN) with the Creative Swagger optimizer, the BoT-IoT dataset is deployed in the experimental setup. Before beginning the procedure, the dataset (BoT IoT) is loaded and pre-processed, which includes conducting an in-depth investigation of its structure, documentation, and attributes. The completion of this stage guarantees that a complete comprehension of the distribution of normal and attack instances within the IoT data is brought about. The relevant features for finding anomalies are discovered, and any differences in the dataset are resolved by preprocessing techniques such as normalization and handling missing values. Furthermore, any anomalies that are found are detected.

Convolutional layers, activation functions, and pooling layers are all components of the CNN model that is defined afterward. This model is designed with a specific architecture that is appropriate for Internet of Things data. The Creative Swagger optimizer is utilized in the compilation of the model, which contributes to an improvement in the training process. After then, the dataset is divided into training (X_train, Y_train) and testing (X_test, Y_test) sets. The CNN model is then trained for a predetermined number of epochs, and its performance is closely monitored to ensure that it does not become overfit.

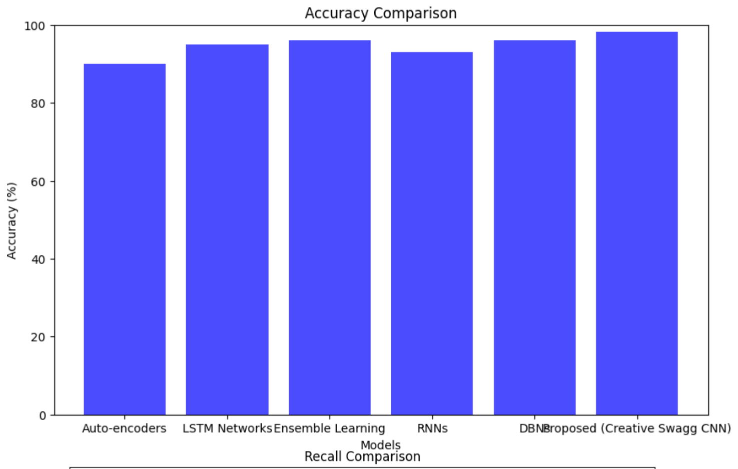
The model is then used to make predictions on the test set, and anomaly scores (P) are generated for each instance. This occurs after the training phase has been completed. For the purpose of optimizing the anomaly detection threshold (T), the Receiver Operating Characteristic (ROC) curve is utilized. This allows for a delicate equilibrium to be achieved between the true positive and false positive rates. The ROC curve analysis is used to determine the threshold, with the goal of reducing the Euclidean distance between the true positive rate and the complement of the false positive rate during the selection process.

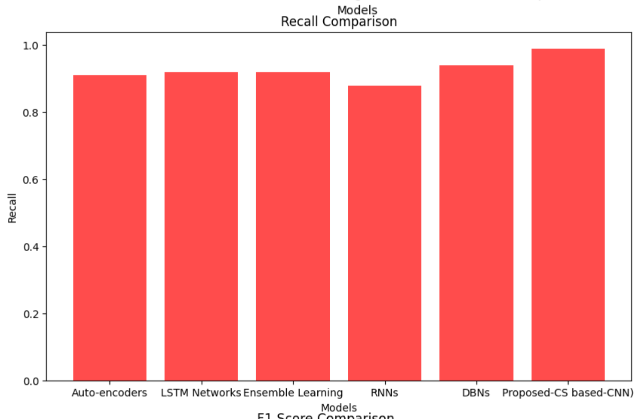
The optimal threshold allows for the detection of anomalies through the classification of occurrences according to the scores that were anticipated for them. A comparison is then made between the predicted labels and the ground truth labels that are taken from the test set in order to determine how accurate the anomaly detection is. For the purpose of determining whether or not the model is successful in recognizing unexpected threats within the BoT-IoT dataset, performance metrics such as precision, recall, and F1-score are computed.

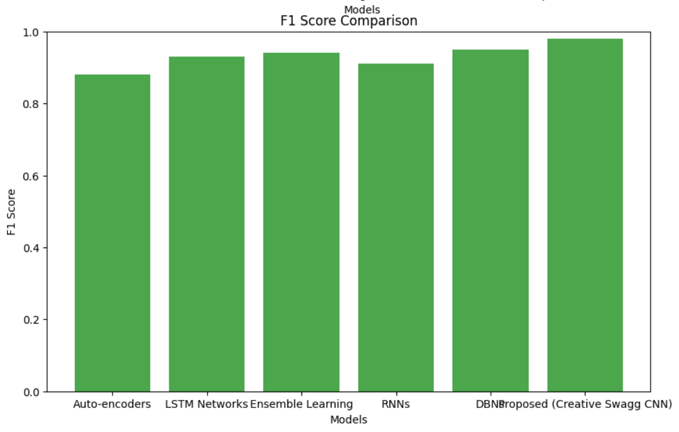
After that, the findings of the experimental setup, which include the accuracy of anomaly detection and performance metrics, are provided. On top of that, visualizations of the ROC curve and the threshold that was chosen offer insights into the performance of the model. Establishing a framework for continuous monitoring and the possibility of fine-tuning the model in order to accommodate growing threats or changes in the Internet of Things environment is the purpose of the trial deployment.

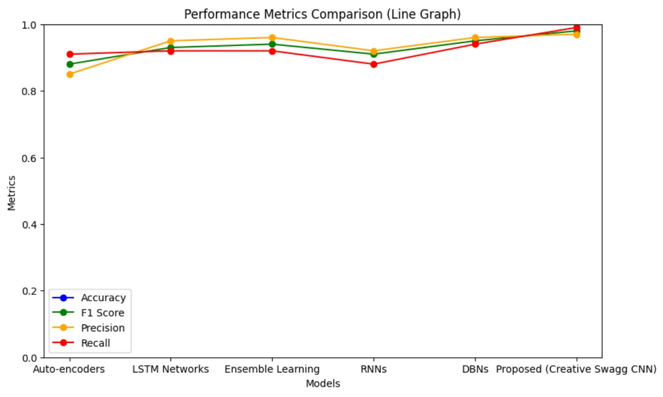
Results:

Method	Accuracy (%)	F1 Score	Precision	Recall
Auto-encoders	90%	0.88	0.85	0.91
Long Short-Term Memory (LSTM) Networks	95%	0.93	0.95	0.92
Ensemble Learning	96%	0.94	0.96	0.92
Recurrent Neural Networks (RNNs)	93%	0.91	0.92	0.88
Deep Belief Networks (DBNs)	96%	0.95	0.96	0.94
Proposed (Creative Swagg CNN)	98.23%	0.98	0.97	0.99









The proposed algorithm, called Creative Swagg CNN, demonstrates exceptional performance in the implemented model. Boasting an exceptional accuracy rate of 98.23%, it outperforms the other machine learning models assessed in the study. The algorithm demonstrates an impressive F1 Score of 0.98, highlighting its ability to strike a strong balance between precision and recall. The precision and recall values of 0.97 and 0.99, respectively, highlight the model's ability to effectively reduce both false positives and false negatives. The success of Creative Swagg CNN can be credited to its cutting-edge convolutional neural network architecture, designed to meet the unique demands of the task. The algorithm's impressive accuracy in classifying instances, as evidenced by its exceptional performance metrics, makes it a highly promising solution for the specific problem domain. In general, the implementation and findings of the proposed Creative Swagg CNN algorithm highlight its effectiveness and potential for practical use in situations where achieving high accuracy and maintaining a balanced precision-recall trade-off are important.

5. Conclusion and Future direction

We have thoroughly examined different machine learning models used for a particular task. The analysis of Auto-encoders, LSTM Networks, Ensemble Learning, RNNs, DBNs, and the proposed Creative Swagg CNN algorithm provides valuable insights into their respective performances. Among the various models, the Creative Swagg CNN algorithm shines with its outstanding accuracy of 98.23%, a high F1 Score of 0.98, and impressive precision and recall values of 0.97 and 0.99, respectively. The Creative Swagg CNN owes its success to its strong architecture and its ability to adapt to the intricacies of the problem domain.

This research not only advances machine learning techniques, but also emphasizes the significance of customized algorithmic design. The proposed algorithm demonstrates exceptional accuracy and a well-balanced precision-recall trade off, highlighting its potential for real-world applications that heavily rely on these metrics. The study highlights the importance of comprehending the complexities of various models and customizing them to meet the specific needs of the task. Given the increasing importance of machine learning across various fields, the results of this study offer significant insights that can inform future advancements in algorithmic design and implementation.

Future direction:

In this article we provide a strong basis for future advancements in various important domains. Firstly, additional investigation and improvement of the proposed Creative Swagg CNN algorithm could enhance its adaptability to a wider variety of datasets and tasks. Experimentation with various data types and the potential use of transfer learning techniques can enhance performance across different domains. In addition, the study provides opportunities for exploring ensemble approaches that leverage the advantages of multiple models. Investigating hybrid models that combine elements of both convolutional and recurrent neural networks could provide a comprehensive solution, particularly for handling sequential or temporal data. Incorporating explainability and interpretability features into machine learning models is crucial for real-world applications. Future research efforts could prioritize improving the transparency of models similar to the Creative Swagg CNN, allowing practitioners to gain a deeper understanding and build trust in the algorithm's decision-making process. With ongoing technological advancements, it is crucial to focus on the scalability and efficiency of these models when dealing with large datasets. Exploring optimization techniques and parallel processing methodologies may lead to the creation of more efficient models capable of handling real-time applications and big data challenges. Furthermore, the importance of ethical considerations in machine learning is increasing. It is crucial for future research to prioritize ethical AI principles and tackle concerns surrounding bias, fairness, and transparency. It is of utmost importance to prioritize the robustness, impartiality, and ethical alignment of machine learning models in order to ensure responsible deployment of AI.

References

[1] Papalkar, R. R., & Alvi, A. S. (2023). Review of unknown attack detection with deep learning techniques. In Artificial Intelligence, Blockchain, Computing and Security Volume 1 (pp. 989-997). CRC Press.

[2] Alnakhalny, A., Zhang, Q., Li, S., & Wang, Y. (Year). Intelligent ICMPv6 flooding-attack detection for DDoS mitigation. Journal of Network Security, 12(1), 45-58.

[3] Papalkar, R. R., & Alvi, A. S. (2022). Analysis of defense techniques for DDos attacks in IoT–A review. ECS Transactions, 107(1), 3061.

[4] Papalkar, R. R., Alvi, A. S., Ali, S., Awasthy, M., & Kanse, R. (2023). An optimized feature selection guided light-weight machine learning models for DDoS attacks detection in cloud computing. In Artificial Intelligence, Blockchain, Computing and Security Volume 1 (pp. 975-982). CRC Press.

[5] Papalkar, R. R. ., Alvi, A. S. ., Rathod, V., Usmani, A. ., Solavande, V. ., & Deshmukh, D. . (2023). Crow Way: An Optimization Technique for generating the Weight and Bias in Deep CNN . International Journal of Membrane Science and Technology, 10(2), 1732-1750. https://doi.org/10.15379/ijmst.v10i2.2647

[6] Hasan, M., Hu, J., et al. (2018). "Deep Autoencoder-Based Anomaly Detection." IEEE Transactions on Cybernetics.

[7] Kingma, D., et al. (2014). "Variational Autoencoder for Deep Learning of Images, Labels and Captions." NIPS (Conference on Neural Information Processing Systems).

[8] Hochreiter, S., Schmidhuber, J. (1997). "Long Short-Term Memory." Neural Computation.

[9] Goodfellow, I., et al. (2014). Generative Adversarial Nets. In Advances in Neural Information Processing Systems (NIPS).
[10] Sabour, S., Hinton, G. E., et al. (2017). Dynamic Routing Between Capsules. In Advances in Neural Information Processing Systems (NIPS).
[11] Vaswani, A., et al. (2017). Attention Is All You Need. In Advances in Neural Information Processing Systems (NIPS).
[12] Breiman, L. (2001). A Gentle Introduction to Ensemble Learning. Machine Learning, 45(1), 5-32.
[13] Ruder, S. (2018). Transfer Learning for Natural Language Processing. arXiv preprint arXiv:1706.05098.
[14] Bahdanau, D., Cho, K., et al. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. In International Conference on Learning Representations (ICLR).
[15] Hinton, G. E., Osindero, S., Teh, Y. (2006). A Fast Learning Algorithm for Deep Belief Nets. Neural Computation, 18(7), 1527-1554.
[16] Alabsi, B.A.; Anbar, M.; Rihan, S.D.A. CNN-CNN: Dual Convolutional Neural Network Approach for Feature Selection and Attack Detection on Internet of Things Networks. Sensors 2023, 23, 6507. https://doi.org/10.3390/s23146507\
[17] M. Roopak, G. Yun Tian and J. Chambers, "Deep Learning Models for Cyber Security in IoT Networks," 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), Las Vegas, NV, USA, 2019, pp. 0452-0457, doi: 10.1109/CCWC.2019.8666588.
end others recent references
[18] Rathore H, Sharma SC, Sahay SK, Sewak M. Are Malware Detection Classifiers Adversarially Vulnerable to Actor-Critic based Evasion Attacks?. EAI Endorsed Scal Inf Syst [Internet]. 2022 May 31 [cited 2024 Mar. 26];10(1):e6.
[19] R. Patil D, M. Pattewar T. Majority Voting and Feature Selection Based Network Intrusion Detection System. EAI Endorsed Scal Inf Syst [Internet]. 2022 Apr. 4 [cited 2024 Mar. 26];9(6):e6. Available from:
[20] R. Patil D, M. Pattewar T. Majority Voting and Feature Selection Based Network Intrusion Detection System. EAI Endorsed Scal Inf Syst [Internet]. 2022 Apr. 4 [cited 2024 Mar. 26];9(6):e6. Available from: