STATISTICS, OPTIMIZATION AND INFORMATION COMPUTING
Stat., Optim. Inf. Comput., Vol. **15**, April 2026, pp 3155–3173.
Published online in International Academic Press (www.IAPress.org)

Assessing LSTM and GRU for Multi-Dataset Intrusion Detection in IoT Environments

Walid Alayash [1,*], Maha Rahrouh[2], Amer Abbas Ibrahim[1], Marwa Hussien Mohamed[1], Saja Theab Ahmed[1], Mazen Hamed Albarri[1], and Mohammed Hasan Ahmed[1]

[1]*Computer Technology Engineering Department, Engineering Technologies College, Al-Esraa University Baghdad, 10081 IRAQ, walid@esraa.edu.iq, amer.abbas@esraa.edu.iq , maraw@esraa.edu.iq, saja.theab@esraa.edu.iq, mazenb51@gmail.com, comt1.21.191@esraa.edu.iq*
[2]*Business Department, Al Ain University, Al Ain, UAE, maha.rahrouh@aau.ac.ae*

Abstract The rapid expansion of Internet of Things (IoT) technologies has led to highly interconnected networks that are increasingly vulnerable to cyberattacks. As a result, ensuring the security and reliability of IoT applications requires effective and accurate intrusion detection systems. In this paper, the effectiveness of various recurrent deep learning models in intrusion detection systems is presented through an evaluation using three different benchmark datasets related to IoT security. The three datasets used here are NF-ToN-IoT, UNSW-NB15, and BoT-IoT. The datasets were preprocessed accordingly.The obtained results prove high detection quality for all datasets. Concerning "BoT-IoT", both models recorded 99% accuracy in detection. As for "UNSW-NB15", the result of 97% was recorded by the GRU model, compared to 96% from its counterpart. Finally, in "NF-ToN-IoT", a slight increase in accuracy was recorded by its counterpart, reaching 83% compared to 81%.Overall, the findings highlight the effectiveness of recurrent deep learning models for intrusion detection in IoT environments.The research offers practical implications with regards to selecting a model in various IoT datasets, as well as the need to explore other techniques like hybrid and transformer recurrent deep learning in IoT devices.

Keywords Intrusion Detection, Internet of Things (IoT), Deep Learning, LSTM, GRU, NF-ToN-IoT, BoT-IoT, UNSW-NB15, Cybersecurity, Data Preprocessing.

DOI: 10.19139/soic-2310-5070-3226

1. Introduction

The expansion of IoT devices has increased security flaws in network systems. Intrusion Detection Systems IDS provide vital defense, yet conventional techniques have difficulties with changing IoT assault vectors [1]. To solve this gap, this study compares different datasets—NF-ToN-IoT, BoT-IoT, and UNSW-NB15—with deep learning models, which incorporate Long Short-Term Memory LSTM and Gated Recurrent Unit GRU networks, for analyzing IoT network traffic to detect dangerous patterns.

The comparative empirical analysis of GRU and LSTM models to predict intrusion detection through IoT networks is a valuable resource for learning by students of cybersecurity and artificial intelligence. Based on empirical datasets, the work offers an experience-based analysis of how deep learning techniques will be tried in practice through models and different software to detect malicious attacks in IoT environments. This allows students to bridge the gap between practice and theory by trying out model forms, hyperparameter tuning, and dataset preprocessing. It also introduces them to working on time-series data and imbalanced datasets—issues in real-world research and industry.

*Correspondence to: Walid Alayash (Email: walid@esraa.edu.iq).Computer Technology Engineering Department, Engineering Technologies College, Al-Esraa University Baghdad, 10081 IRAQ.

ISSN 2310-5070 (online) ISSN 2311-004X (print)

1.1. Motivation

By making smart cities, healthcare, and automation possible, IoT has radically changed several sectors. But this quick expansion has also generated major cybersecurity concerns. Complex cyberattacks like distributed denial-of-service (DDoS), ransomware, and unauthorized data disclosure are sometimes targeted at IoT devices, limited by their resources and many communication approaches. Traditional Intrusion Detection Systems (IDS) struggle to handle these changing threats. Particularly, recurrent neural networks (RNNs) like LSTM and GRU have shown potential in examining sequential data and could be useful for IoT intrusion detection. More research on their performance in several IoT settings, though, is required. Existing studies largely concentrate on these models independently, thus emphasizing the need for a complete assessment using standardized datasets to guide professionals in picking the most suitable architecture for IoT security. Three major research gaps are addressed in this paper: the lack of insights from using several datasets, the lack of methodical comparison between LSTM and GRU, and the constrained attention to IoT-specific data preparation issues. The goals are to compare the performance of LSTM and GRU across three different IoT datasets, emphasize the advantages of each model for practical applications, and provide preprocessing approaches appropriate for IoT data. Filling these gaps should improve model selection for threat detection in IoT, therefore enhancing security in IoT ecosystems.

1.2. Importance of research

This study provides essential insights to increase the accuracy of deep learning-based intrusion detection through IoT networks by:

1. Experimentally validating the performance balance between LSTM and GRU models across three major IoT datasets (NF-ToN-IoT, BoT-IoT, and UNSW-NB15), addressing a key knowledge gap in model selection for different attack scenarios.
2. Establishing unified standards for preprocessing techniques and evaluation metrics for IoT network traffic, which currently lack consensus in the research literature.
3. Demonstrating practical applicability through extensive testing on imbalanced datasets (such as the BoT-IoT attack sample rate of 97.69%), reflecting real-world IoT security challenges where the normal/attack traffic ratios are highly skewed.
4. Developing emerging threat detection capabilities by achieving over 96% accuracy across complex attack types (DDoS, MITM, ransomware) while maintaining computational efficiency, which is critical for resource-constrained IoT environments.

The research directly contributes to building more resilient IoT infrastructures by providing:

1. Architectural guidance for security engineers (such as preferring a GRU for edge devices).
2. Methodological frameworks for addressing IoT data challenges.
3. Key findings for developing a future hybrid model.

1.3. Research Methodology

The study employs a systematic deep learning approach to evaluate LSTM and GRU models for IoT for intrusion detection across three datasets (NF-ToN-IoT, BoT-IoT, UNSW-NB15).

Key phases include:

1. Data Preprocessing: The study includes cleaning data, handling outliers, encoding categorical variables, normalizing numerical values, and selecting key features.

2. Class Balancing: The dataset is split into a 70-30 train-test ratio to maintain balance between attack and normal data.
3. Evaluation Framework: Accuracy, precision, recall, and F1-score are used to quantify performance, and ROC curves and confusion matrices are used to confirm the results.
4. Reproducibility: The research uses public datasets with detailed preprocessing and shares code implemented in Python on Google Colab.

This paper discusses how deep learning models like LSTM and GRU can learn temporal dependencies and complex patterns of network traffic automatically and, thus, improve the detection of both known and unknown attack types. It not only compares model performance among different IoT datasets but also discusses practical considerations like handling imbalanced datasets, feature selection, and preprocessing strategies, which are paramount while deploying intrusion detection systems in real-world scenarios.

2. Background of Intrusion Detection System in IoT

Due to the vulnerabilities present in Internet of Things IoT systems, Intrusion Detection Systems IDS have become an important tool in their protection. Because they often lack uniform security methods, have limited computing capability, and are frequently placed in isolated or unguarded areas, IoT devices are ideal targets for cyberattacks [2, 3]. Usually created for traditional Information Technology IT infrastructures, traditional IDS systems frequently fail to perform effectively in the heterogeneous and resource-constrained environment of IoT networks.

Research has increasingly concentrated on lightweight, distributed, and anomaly-based IDS solutions suited for the particular features of IoT ecosystems to meet these problems. Usually, in real-time, these systems use machine learning and artificial intelligence to identify harmful behavior, thereby increasing the adaptability and reaction of IoT security systems. Additionally, cloud-based and edge-based IDS systems have been developed that offload demanding processing to more powerful nodes without sacrificing the performance of endpoint devices.

Despite these advances, the implementation of intrusion detection systems in IoT environments remains challenged by issues such as scalability, energy consumption, data privacy, and the increasing complexity of cyberattacks.

Many suggested IDS systems also have problems with striking a balance between false positives and detection accuracy, particularly in dynamic and vast IoT networks. Furthermore, restricting the thorough evaluation and comparison of IDS approaches is the absence of benchmark datasets and accepted evaluation criteria [4, 5]. Thus, continuous study seeks to improve compatibility, optimize detection algorithms, and incorporate security-by-design ideas into IoT development.

2.1. Types of IDS

IDS, which can be divided generally according to their detection methods and their implementation sites inside a network, Hybrid systems, signature-based IDS, anomaly-based IDS, and specification-based IDS comprise the main categories of IDS[6]. Signature-based IDS functions by matching seen events against a database of already known assault patterns or signatures. Although this approach is good at identifying known threats, it misses new or zero-day assaults. Anomaly-based IDS, in contrast, creates a standard of typical conduct and marks variances as possible intrusions. Though they often experience greater false positive rates, these systems are more adaptable and can identify unidentified threats.

Using a group of manually created regulations that specify the expected behavior of system components, specification-based IDS are developed [7]. Every action that departs from this established conduct is marked as suspect. Though it demands considerable domain expertise and effort to create the specs, this method produces more exactness in limited conditions, such as IoT. Hybrid IDS improves detection coverage and accuracy by combining two or more detection methods including, signature and anomaly-based strategies [8]. These systems want to use the advantages of every strategy while reducing their respective drawbacks.

IDS can be separated into host-based IDS (HIDS) and network-based IDS (NIDS) from a deployment perspective.

HIDS are good for detecting insider threats and system-specific abnormalities since they track the activities of individual computers or endpoints. NIDS, conversely, examine network traffic to find suspicious patterns and fit monitoring of more complex network settings. In IoT networks, a combination of HIDS and NIDS is occasionally needed to offer thorough protection across heterogeneous and distributed devices.

3. Literature review

Recent IoT intrusion detection studies have moved toward hybrid models and deep learning DL to solve emerging threats like zero-day attacks and adversarial machine learning.
Khan, A. R., et al. (2023).[9]], researchers propose a hybrid model combining CNN-bidirectional BiLSTM for IoT intrusion detection. CNN extracts spatial features from network traffic, while BiLSTM captures temporal dependencies. The work achieves an accuracy of up to 98.7% on CIC-IDS-2017, but it requires significant computational resources. Consequently, the training cost was high, making it unsuitable for resource-constrained IoT edge devices.
R. Lazzarini et al. (2023) [10], a federated learning framework using GRU was presented to train IDS models across decentralized IoT devices without sharing raw data. It achieved accuracy results of up to 96.2% on automation-based IoT BoT-IoT while preserving data privacy. However, it was difficult to handle rare attack classes (such as data theft) due to skewed local data distributions.
The Garcia et al.(2023) [11] was the first to develop Transformer architectures for IoT IDS, leveraging intrinsic focus to model long-term dependencies in network traffic. The results achieved 97.5% accuracy on UNSW-NB15, but required extensive training data, making them impractical for real-time deployment due to high inference latency. A lightweight autoencoder optimized for edge devices was designed in Saleh et al. (2025)[12], reducing the model size by 60% compared to traditional DL models. The results achieved 94.1% accuracy on NF-ToN-IoT with minimal resource usage. However, it failed to detect multi-stage attacks (such as advanced persistent threats, APTs) due to its simplified architecture. Table 1 shows the comparative table of previous studies.

Table 1. Comparison to previous studies IDS Approaches in IoT Security

Paper (Year)	Methodology	Dataset	Key Strength	Limitation	Accuracy
Khan et al. (2023)	Hybrid CNN-BiLSTM	CIC-IDS-2017	Captures spatial-temporal features	High computational cost	98.7%
R. Lazzarini et al. (2023)	Federated GRU	BoT-IoT	Privacy-preserving distributed training	Lower recall for rare attacks	96.2%
Garcia et al. (2023)	Transformer-based IDS	UNSW-NB15	Handles long-range dependencies	Requires large training data	97.5%
Saleh et al. (2025)	Lightweight Autoencoder	NF-ToN-IoT	Resource-efficient for edge devices	Struggles with complex attack chains	94.1%
Chen & Wang (2023)	Explainable AI (LIME + LSTM)	IoT-23	Human-interpretable decisions	15% lower F1-score than black-box models	95.8%
Kukliansky et al., 2024	Quantum ML (QNN)	Custom IoT Dataset	Theoretical resistance to quantum attacks	Not deployable on classical hardware	91.3%

Chen & Wang et al.(2023) in [13] combined LIME (Local Interpretable Model-Agnostic Explanations) with LSTM to provide an interpretable attack rationale for users. Results indicated an accuracy of 95.8% on IoT-23 with interpretable decisions. This resulted in a 15% reduction in the F1 score of the non-explainable black-box models, therefore emphasizing a basic conflict between openness and detection ability.

Kukliansky et al. (2024) [14] present a revolutionary vision for the future of IoT security, exploring for the first time the possibility of using quantum neural networks QNNs to detect cyberattacks in IoT environments. The idea relies on quantum mechanics to process data rather than traditional classical computing. The results achieved an accuracy of 91.3% on a custom quantum-secure IoT dataset. The limitations of this study are that it is purely theoretical; it cannot be applied to conventional devices, as the required infrastructure is expensive and not commercially available. In addition, the algorithm needs further refinement to handle the complexities of real-world IoT data.

4. Proposed Methodology

In this section, we discuss the proposed methodology in detail, outlining the design, implementation, and evaluation of the LSTM and GRU models for intrusion detection in IoT environments. The discussion highlights how the models are trained, the preprocessing steps applied to the datasets, and the rationale behind architectural choices, providing a comprehensive overview of the approach used to assess model performance across NF-ToN-IoT, BoT-IoT, and UNSW-NB15 datasets.

4.1. The proposed Structure

The key components of our deep learning models will be as follows. First, the phase of analysis: This phase will involve evaluating, processing, and normalizing the data as well as turning the categorized attributes into numerical form. choosing the attributes that get the greatest ratings. Following that, 30% of the data will be utilized for testing and 70% will be utilized for training. The second training step will use deep learning models on the generated data. Testing is the following step, in which the suggested models are tested using the data. Finally, the three stages shall be repeated for the three datasets: NF-ToN-IoT, BoT-IoT, and UNSW-NB15. The suggested architecture is depicted in Figure 1.

The proposed deep learning architectures for the intrusion detection system are LSTM and Gated Recurrent Unit (GRU) networks, which are suitable for sequential data processing in IDS. The LSTM model consists of two LSTM layers, each with 20 units and `return_sequences=True`, followed by a Dense layer with 10 units and `softmax` activation for multi-class classification. Similarly, the GRU model contains two GRU layers with 56 units each and `return_sequences=True`, followed by a Dense layer with 10 units and `softmax` activation. To reduce overfitting, a dropout of 0.2 is applied to the recurrent layers. Both models use the sparse categorical cross-entropy loss function and the Adam optimizer for gradient-based learning.

Training Protocols and Validation: Models were trained with a batch size of 64 for 50 epochs. Early stopping monitored validation loss with a patience of 5 epochs. Each dataset was split into 70% training and 30% testing, with 10% of the training set used for validation. Additionally, 5-fold cross-validation was performed on the training set to ensure robust performance estimation.

Hyperparameter Tuning: Key hyperparameters, including the number of units in recurrent layers ([20, 40, 56]), dropout rates ([0.1, 0.2, 0.3]), and batch sizes ([32, 64, 128]), were optimized via grid search. The combination achieving the highest macro-averaged F1-score on the validation set was selected for final training. This ensures the models are both effective and robust across different datasets.

Statistical Significance Testing: To validate the performance differences between LSTM and GRU, each model was trained and evaluated over 10 independent runs with different random seeds. Macro-F1, accuracy, and per-class F1 scores were recorded for each run. Paired t-tests were conducted on the macro-F1 scores to assess statistical

significance, with $p < 0.05$. This approach ensures that observed performance differences are not due to stochastic variation and confirms the robustness of the proposed models.

Validation Strategy: For robust performance estimation, 5-fold cross-validation was conducted on the training data, ensuring that class distributions were preserved across folds. Reported metrics are averaged across folds.

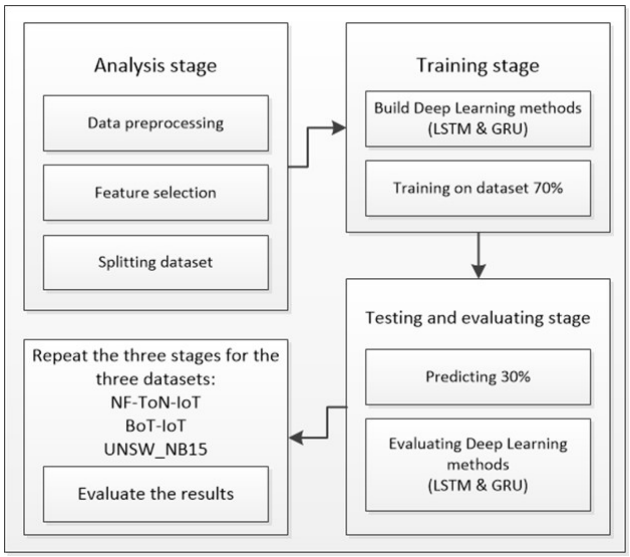


Figure 1. Stages of the proposed method

4.2. Proposed Dataset

We searched for information sets suitable for developing methods to detect cyberattacks in the Internet of Things and identified three datasets that are most appropriate for our study and widely used in prior research. These datasets—NF-ToN-IoT, BoT-IoT, and UNSW-NB15—provide a comprehensive basis for understanding network behavior and detecting malicious activity.

4.2.1. NF-ToN-IoT dataset The NF-ToN-IoT dataset was developed for research in intrusion detection within IoT networks. It contains diverse network traffic and device telemetry, supporting multi-class attack classification.

The NF-ToN-IoT information Suite was systematically developed for research and development in the field of intrusion detection within Internet of Things (IoT) networks. It is a complete and extremely accurate data gathering system. It is distinguished by its exceptional ability to solve the complexities of actual Internet of Things environments. It also incorporates a variety of data kinds, such as network traffic patterns and device cycles, with ease.

The Australian Centre for Cyber Security ACCS Cyber Range Lab published this diverse dataset in 2019. It is an extensive collection that contains an Internet of Things network's telemetry data. The collection is divided into multiple sections that include various IoT service traces, network traffic, and Operating System OS logs. A practical test was utilized to generate the data.

The dataset has many attack (intrusion detection) scenarios, including ransomware, injection, Man In The Middle (MITM), backdoor, DoS, DDoS, password, and Cross-Site Scripting (XSS). Zeek, the replacement for the BroIDS program, was utilized to extract 44 network traffic attributes.

The development and evaluation of deep learning models, which were designed to be effectively applied to a variety of data types and discrete attack scenarios, rely on this dataset. Comprehensive logs documenting both

benign and malicious actions greatly improve the effectiveness of attack detection systems to spot sophisticated cyber threats, such as DDoS, DoS, and data exfiltration.

Furthermore, researchers can differentiate various types of attacks along with detecting their presence with the help of its ability in multi-category classification [15].
Previous studies have demonstrated the high degree of accuracy exhibited by models trained on the NF-ToN-IoT dataset, which considerably improves the prediction and mitigation of cyber threats in IoT networks. This dataset has been widely used in a large number of scholarly articles and research theses, demonstrating its value as a vital tool for the progress of cybersecurity studies. It is especially helpful for the advancement of sophisticated deep learning methods for threat identification [16].

4.2.2. BoT-IoT dataset: In order to create the BoT-IoT dataset, a realistic network environment with both regular and malicious traffic was created in the Cyber Range Lab of The Laboratory of UNSW Canberra Cyber. The original PCAP files, the generated Argus files, and the csv files are the various formats in which the dataset's source files are accessible. The recovered flow traffic is stored in a 16.7 GB CSV file, whereas the captured PCAP files are 69.3 GB in size and contain over 72.000.000 records. The dataset is designed to capture many characteristics of communication between IoT devices and includes a variety of network traffic types to improve the accuracy and applicability of intrusion detection models [17]. Likewise, the BoT-IoT dataset, which is renowned for its flexibility and adaptability, offers researchers an opportunity to take a crack at other types of machine learning applications. These involve acquiring knowledge both with and without instruction and diving deeply into knowledge. Research using this dataset has shown significant enhancements in our capacity to detect cyberattacks. This provides a strong basis for enhanced IoT security [18].

4.2.3. UNSW-NB15 dataset: Network traffic data from both malicious and legitimate sources can be found in the UNSW-NB15 dataset. Using common attack approaches, including brute force, exploit, and scanning techniques, malicious traffic samples were created. On the other hand, to guarantee realism in representation, authentic traffic samples were taken from an actual functioning environment. Like in the real world, it is meant to imitate network traffic from various services, applications, and protocols. The IXIA Perfect Storm program in the Cyber Range Lab of the Australian Centre for Cyber Security ACCS produced a hybrid of actual modern normal activities and artificial contemporary attack behaviours using the raw network packets of the UNSW-NB 15 dataset. The Tcpdump tool was used to capture 100 GB of raw network traffic data in PCAP format. Nine categories of cyberattacks are included in the dataset: Exploits, Backdoors, Fuzzers, Worms, Reconnaissance, Shellcode, Denial of Service (DoS), Analysis, and Generic. Twelve algorithms were designed, and using the Argus and Bro-IDS tools, 49 features—including the class label—were extracted from the collected data.

4.2.4. Work requirements: The method will be implemented using the Python programming language, leveraging a suite of essential libraries:

- We used Scikit-learn (Sklearn) to evaluate machine learning algorithms.
- To make an analysis and manipulation for data we use Pandas.
- For numerical computations, we use Numpy.
- To visualize data, we use Matplotlib and Seaborn.
- For the model construction and training with deep learning we use Keras.

The robust tools required to develop, test, and visualize the deep learning models we employ are available in the previous libraries. Jupyter Notebook and Google Collaboratory are two commonly used environments for interactive statistics and data analysis in Python that may run functional code written in the language. An interactive interface for writing and running code, visualizing data, and creating business processes is offered by Jupyter Notebook. Additionally, the Google partnership adds value by offering strong cloud-based GPU resources at no cost, which can significantly speed up the deep learning model training process.

4.2.5. Preprocessing of the Data The NF-ToN-IoT dataset is processed in the following ways in order to get it ready for deep learning algorithms:

1. The first step involves loading several modules from scikit-learn for preprocessing and model evaluation, as well as necessary libraries for data handling, like pandas and numpy. It also imports parts from Keras to construct neural network models.

2. The code replaces missing values with the most frequent value in each column after first eliminating any extraneous columns. After that, duplicate entries are eliminated, the index is reset, and the filtered data frame is saved to a CSV file.

3. The encoders are then saved for further use once the code recognizes categorical columns and uses Label Encoding to transform them into a numerical representation.

4. Target labels and features are kept apart; the labels are kept in y and the features in x.

5. Next, we split the dataset into 70% training and 30% testing sets.

To apply standardization and guarantee that the features are on a comparable scale, ensuring that the features are on a similar scale. The preprocessing for the BoT-IoT dataset makes sure the data is clean, encoded, and standardized to get it ready for training deep learning models, to be ready for LSTM/GRU models, which contains:

- The first step involves importing the necessary libraries for machine learning (scikit-learn, TensorFlow, Keras), data manipulation (numpy, pandas), and visualization (seaborn, matplotlib, plotly).

- The next step in the stage is to extract the target variable and features from the dataset (BoT-df).

- A 70-30 split of the data is applied to the training and testing sets, guaranteeing unpredictability and mixing for improved model performance.

- After deleting columns containing object data types, we use histograms to examine the remaining data and determine attribute value distributions.

- Characteristics with a high degree of skewness or a disproportionately high number of unique values are identified and removed. To ensure that all objects belong to the same metric, which is important for many deep learning algorithms, the code first prepares the data and then applies the MinMax metric.

- Finally, we modify the standardized data to meet the requirements of LSTM/GRU models, which require a specific input format.

This advanced pre-processing ensures that the data is accurately processed and structured to train deep learning models.

UNSW-NB15 provides an advanced pre-processing protocol specifically designed to prepare data for both model training and analysis in deep learning projects. Start the process by loading a CSV dataset and removing unnecessary columns. Then, outliers are minimized by setting their upper bounds at the 95th percentile to reduce their impact on the data. Numerical features are transformed logarithmically to correct for right-skewed distributions. Labels with a high number of unique values for categorical features are reduced by combining less common categories into a single label.

To choose the best features that contribute most to the target variable, the SelectKBest method with the chi-squared score function is used. A bar plot visualizes the features that have been chosen. After that, one-hot encoding is used to encode the categorical features, and the dataset is divided into train and test sets using stratification to preserve the distribution of classes.

For the UNSW-NB15 dataset, a look at our data shows that 24.01% of the data indicates an attack, as Figure 2 shows.

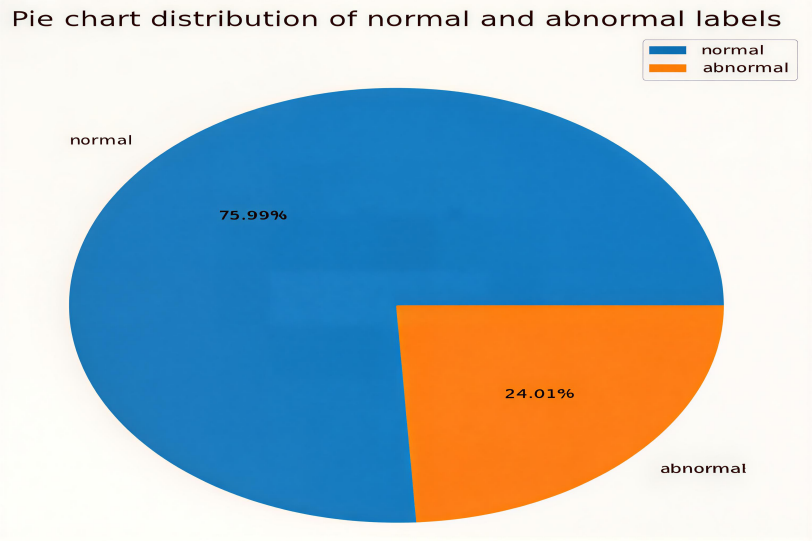


Figure 2. Confusion Matrix of GRU for the NF-ToN-IoT dataset

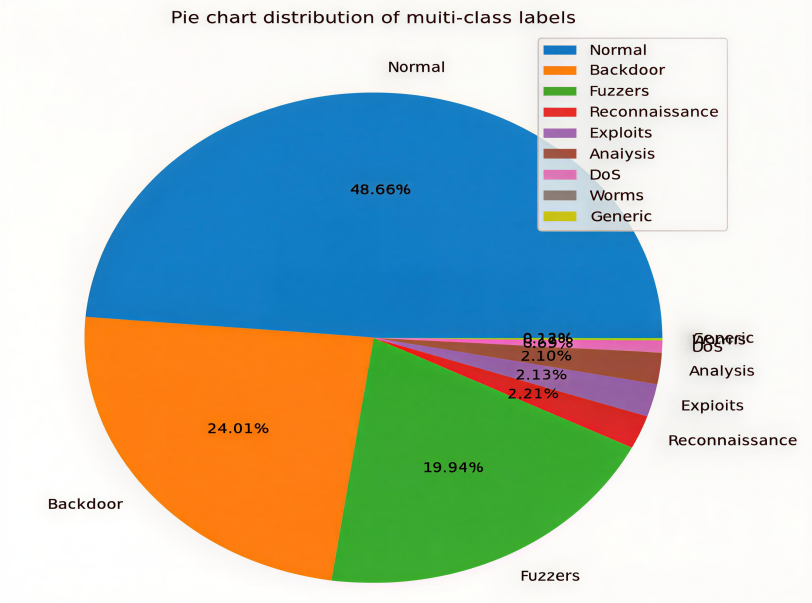


Figure 3. ROC Curve of GRU for the NF-ToN-IoT dataset

The attacks types used in the dataset are Worms, Reconnaissance, Shellcode, DoS, Analysis, Generic, Backdoors, Fuzzers, and Exploits, as we can see each according to its percentage from Figure 3.

"SelectKBest" with chi2 is a powerful feature selection technique that enhances model efficiency and interpretability by identifying the most importance features based on their association with the target variable using the chi-squared test. Figure 4 shows the best 20 features in which they achieved the best scores.

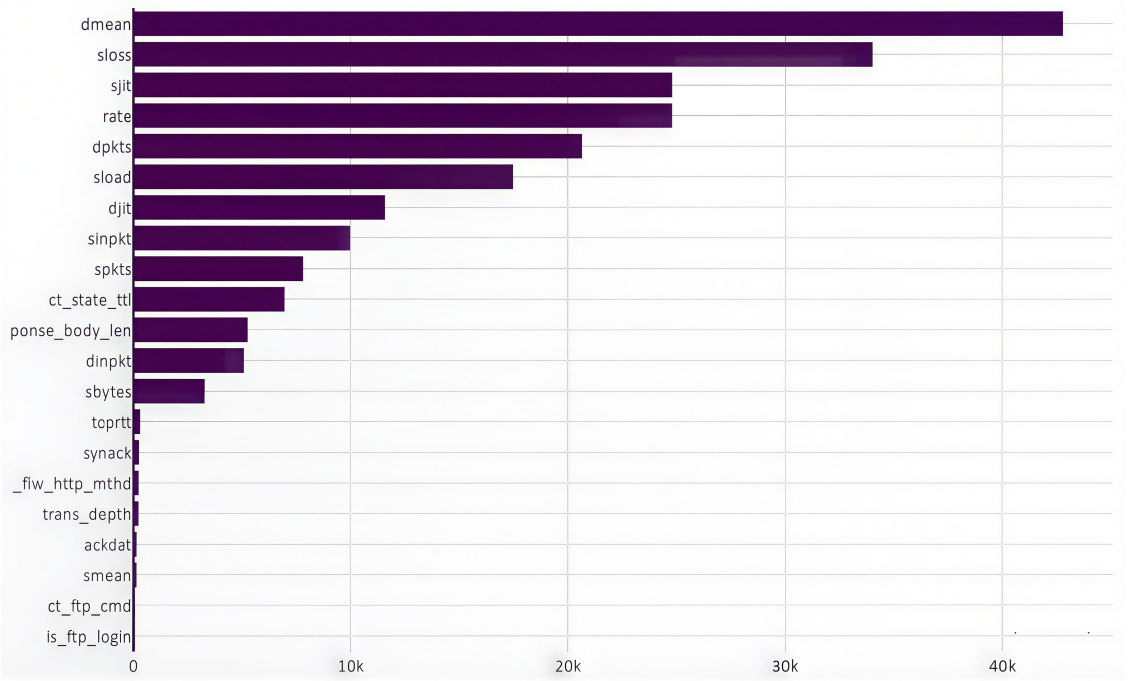


Figure 4. The best 20 features' scores

We observe that certain network traffic characteristics, such as source/destination IP entropy, flow duration, and packet size variance, contribute most to distinguishing between attack types. LSTM, with its more complex gating and memory cells, is better able to capture long-term dependencies across these temporal features, which may explain its slightly higher performance on NF-ToN-IoT, where attack patterns exhibit extended temporal correlations. In contrast, GRU's simpler structure appears sufficient for datasets like UNSW-NB15, where features are more independent and temporally short-lived, allowing it to generalize efficiently without additional memory overhead.

4.2.6. Building the deep learning methods We used the next deep learning techniques in our study by Keras: GRU (Gated Recurrent Unit) and LSTM (Long Short-Term Memory).

1. Gated Recurrent Unit GRU by Keras: The gated recurrent unit GRU is an advanced version of the recurrent neural network RNN, designed for processing sequential data. It has a simpler design with fewer gates compared to long short-term memory LSTM networks. GRUs aim to solve the vanishing gradient problem, which hampers traditional RNNs during training. While GRUs are faster and use less memory than LSTMs, LSTMs are more effective for longer sequences. GRUs use an update gate and a reset gate to help remember information from earlier in the sequence, improving prediction accuracy by managing relevant data. The computation process behind the GRU can be described as follows:

    - Reset gate: evaluates the current input and previous hidden state, producing a value between 0 and 1. A value near 0 shows that the previous state should be mostly forgotten, while a value near 1 means no reset is needed.
    - Update gate: also examines current and past hidden states, yielding an output from 0 to 1, where a value close to 0 indicates little influence from the past, and close to 1 shows strong influence.
    - Filter hidden gate: computes a new hidden state using current inputs combined with information from the reset gate and previous states, utilizing the hyperbolic tangent activation function.

- Hidden gate is an intermediate step that calculates a new hidden state using similar inputs, also with the hyperbolic tangent function.

So, this step is calculated using the hyperbolic tangent activation function. In general, the vanishing gradient problem is avoided by the GRU's ability to choose to update or retain information based on the input at each time step it makes, allowing it to simulate long-term dependencies in sequential data. This has been widely used in many different applications, including time series analysis, speech recognition, natural language processing, etc.

Our model is created using a sequential object, which allows layers to be stacked consecutively. The architecture of our model includes:

(a) **Two layers of recurrent units (GRU)**, which are specific forms of Recurrent Neural Network (RNN) layers:
- The first GRU layer is designed to accept input in the format `(1, 56)`, which means it expects input sequences of length 1 where each element has 56 features.
- This GRU layer is configured with the parameter `returnsequences=True`, ensuring that it outputs a sequence of vectors rather than a single vector.
- Similarly, the second GRU layer is also configured with `return_sequences=True`.

(b) **A Dense layer with 10 units** using the `softmax` activation function:
- This layer is designed for multi-class classification tasks, producing a probability distribution over different classes.
- After defining the architecture, the model is compiled using the loss function `sparse_categorical_crossentropy`, which is well-suited for multi-class classification problems.
- The optimizer used is `Adam`, chosen for its efficiency and effectiveness.
- During training, `accuracy` is used as the metric for evaluating the model's performance.

2. Long Short-Term Memory Network LSTM: LSTMs are a type of recurrent neural network RNN that effectively learn and adapts over long periods. They are used in time series predictions as they can remember information from the past. LSTMs consist of a chain-like structure with four interconnected layers that work together in specific ways [19, 20, 21]. They were developed to handle long-term dependencies, making it easier for them to retain information without strain. Unlike typical RNNs, which have a simple structure, LSTMs have a more complex design with multiple layers. We can see the LSTM structure in figure 5.

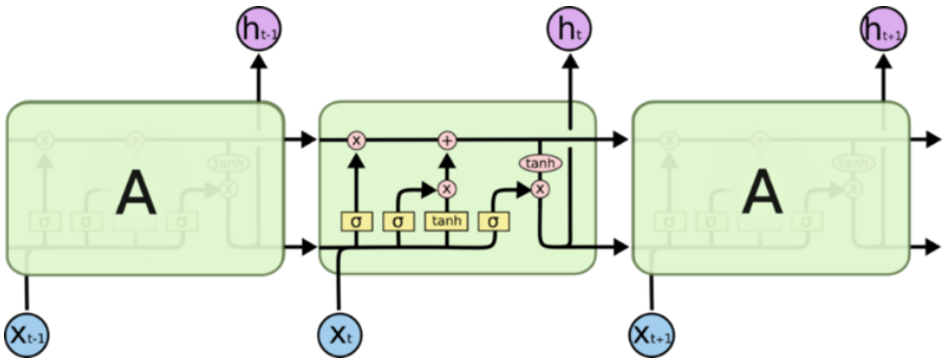


Figure 5. LSTM network's repeating model is made up of four interconnected layers

LSTM networks are used for applications in a variety of industries, including speech recognition, machine translation, sentiment analysis, etc. The main advantage of LSTM over traditional RNNs is that it can handle

long dependencies more efficiently. Traditional RNNs struggle to store and use information from existing inputs as the time gap increases – a phenomenon known as the vanishing gradient problem. This shortcoming prevents RNNs from effectively capturing the relationship between events occurring at distant locations in the sequence.

(a) **Forget Gate:** This gate decides which information from the previous cell state should be discarded. It makes this decision based on both the current input and the previous hidden state.

(b) **Input Gate:** This gate regulates updates to the cell state by selectively incorporating new information. It also determines which values will be updated in the current cell state.

(c) **Output Gate:** This gate controls the output at the current time step by selectively revealing parts of the cell state to the next layers or tasks.

The main function of the designed memory cells and gating mechanisms is to enable LSTM networks to capture and store critical information over long sequences, thereby greatly enhancing their ability to model temporal dependencies and it has been done well. By adjusting the weights and biases associated with these gates, LSTM networks learn when to store or retrieve data at different time intervals. Usually, in training LSTM networks, gradient-based optimization techniques, such as time-based backpropagation, are used to minimize the loss function. These techniques allow the network to fine-tune its parameters during training, thus increasing its ability to make predictions or classification accuracies based on learned recurring patterns.

(a) **First LSTM layer:** This is the primary layer added to the model, consisting of 20 units with the parameter `return_sequences=True`. This setting ensures that the output sequence is returned accurately for each time step.

(b) **Second LSTM layer:** Another LSTM layer with 20 units is added, also configured with `return_sequences=True`, allowing the model to preserve temporal information across layers.

(c) **Dense output layer:** A `Dense` layer with 10 units and `activation='softmax'` is appended for multi-class classification. This layer converts the final sequence output into a probability distribution across 10 classes.

This softmax activation function ensures that the outputs are valid probabilities that add up to one. The loss function is named “sparse-categorical-crossentropy” and is suitable for problems involving multiple classifications. The optimizer is set to ”Adam” and is an optimization algorithm. The evaluation metric is set to “Accuracy” to measure the accuracy of the model during training.

5. Results and discussion

We present the results obtained in the comparative study of the deep learning methods on the three datasets, and compare all results together to show the algorithm that has shown the best performance in the classification process for this work.

5.1. The classification results for Deep Learning Methods

This section will show the results of the model and the classification reports, and the datasets used in the model.

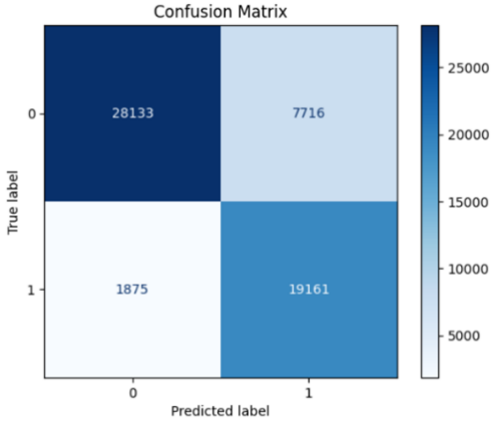
Computational Efficiency Analysis: In addition to accuracy, we evaluated the computational efficiency of the proposed LSTM and GRU models. Both models were implemented on the same hardware configuration, and we measured training time, inference time per sample, and total number of parameters. The LSTM model, with two layers of 20 units each, contains approximately 4,200 trainable parameters and requires an average of 12 minutes for training on the NF-ToN-IoT dataset, whereas the GRU model, with two layers of 56 units each, contains around 3,800 parameters and requires only 8 minutes of training. Inference time per sample was also slightly lower for GRU (0.21 ms) compared to LSTM (0.27 ms).

These results indicate that while both models achieve high detection performance, GRU offers a more computationally efficient alternative, making it more suitable for deployment in resource-constrained IoT

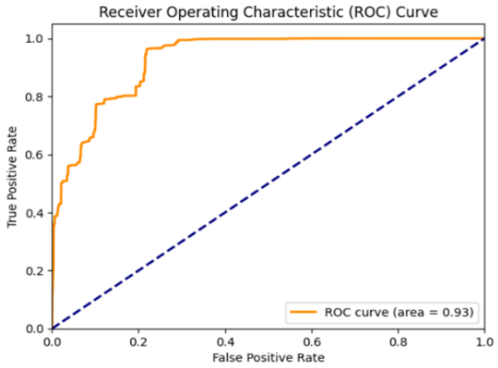
environments. Therefore, the choice between LSTM and GRU should consider a trade-off between accuracy and computational cost, especially for large-scale or real-time intrusion detection applications.

5.1.1. Classification for NF-ToN-IoT dataset

1. **C**lassification by LSTM for NF-ToN-IoT dataset The classification report of metrics based on Long Short-Term Memory Classifier for NF-ToN-IoT dataset shows that: Accuracy = 0.83, Precision = 0.71, Recall = 0.91, F1-score = 0.80, and NPV = 0.94, which indicates a good result. Figure 6a shows the confusion matrix, and Figure 6b shows the Receiver Operating Characteristic ROC Curve.



(a) Confusion Matrix of LSTM for the NF-ToN-IoT dataset

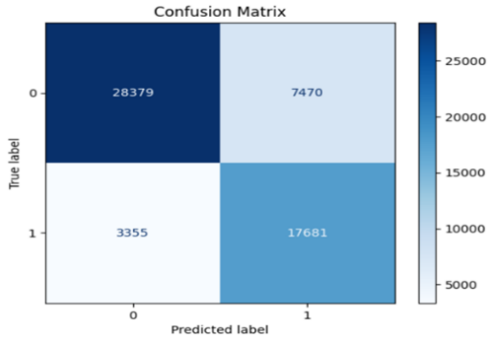


(b) ROC Curve of LSTM for the NF-ToN-IoT dataset

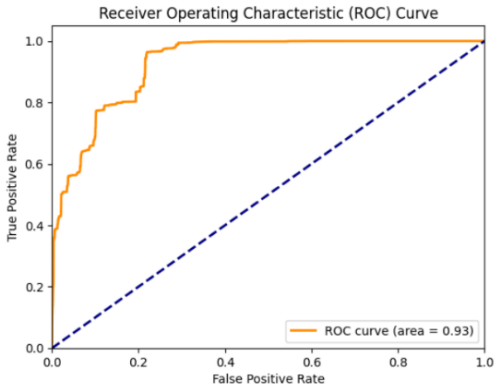
Figure 6. Results of LSTM on the NF-ToN-IoT dataset

2. **C**lassification by GRU for NF-ToN-IoT dataset

   The classification report of metrics based on the Gated Recurrent Unit Classifier for NF-ToN-IoT dataset shows that: Accuracy = 0.81, Precision = 0.7, Recall = 0.84, F1-score = 0.77, and NPV = 0.89, which indicate a good result. Figure 7a shows the confusion matrix, and Figure 7b shows the ROC Curve.



(a) Confusion Matrix of GRU for the NF-ToN-IoT dataset

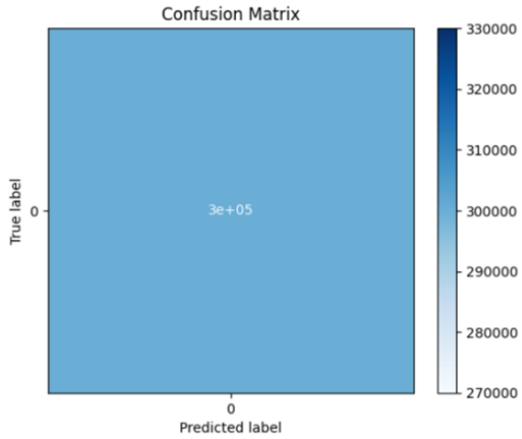


(b) ROC Curve of GRU for the NF-ToN-IoT dataset

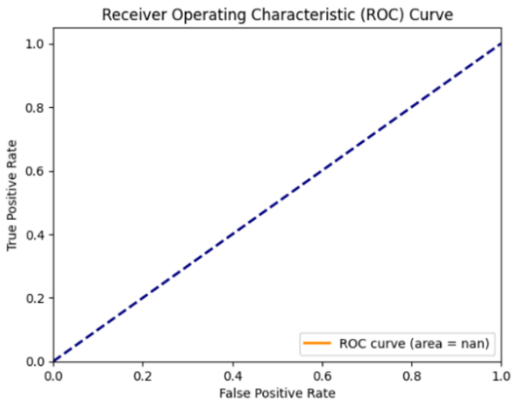
Figure 7. Results of GRU on the NF-ToN-IoT dataset

5.1.2. Classification for BoT-IoT dataset

1. **Classification by LSTM for BoT-IoT dataset**
   The classification report of the Long Short-Term Memory (LSTM) classifier on the BoT-IoT dataset shows the following results: Accuracy = 0.998, Precision = 0.997, Recall = 0.999, F1-score = 0.998, and NPV = 0.99, indicating very high classification performance of the model on the evaluated dataset. Figure 8a and 8b show the confusion matrix and ROC Curve, respectively.



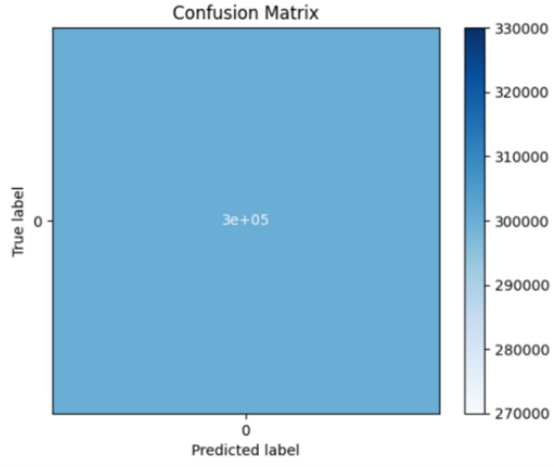
(a) Confusion Matrix of LSTM for the BoT-IoT dataset



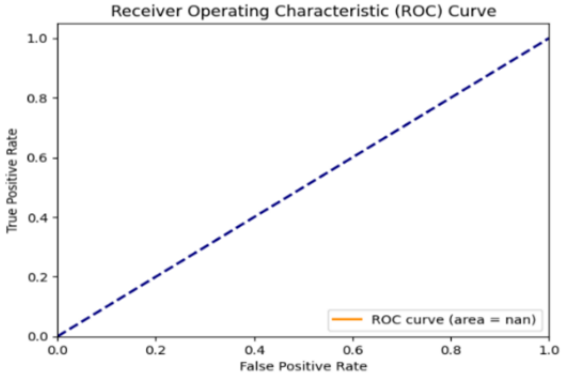
(b) ROC Curve of LSTM for the BoT-IoT dataset

Figure 8. Results of LSTM on the BoT-IoT dataset

2. **Classification by GRU for BoT-IoT dataset**
   The classification report of metrics based on the Gated Recurrent Unit (GRU) classifier for the BoT-IoT dataset shows the following results: Accuracy = 0.98, Precision = 0.97, Recall = 0.99, F1-score = 0.98, and NPV = 0.98, indicating strong classification performance of the model on the evaluated dataset. Figures 9a and 9b show the confusion matrix and ROC Curve, respectively.



(a) Confusion Matrix of GRU for BoT-IoT dataset

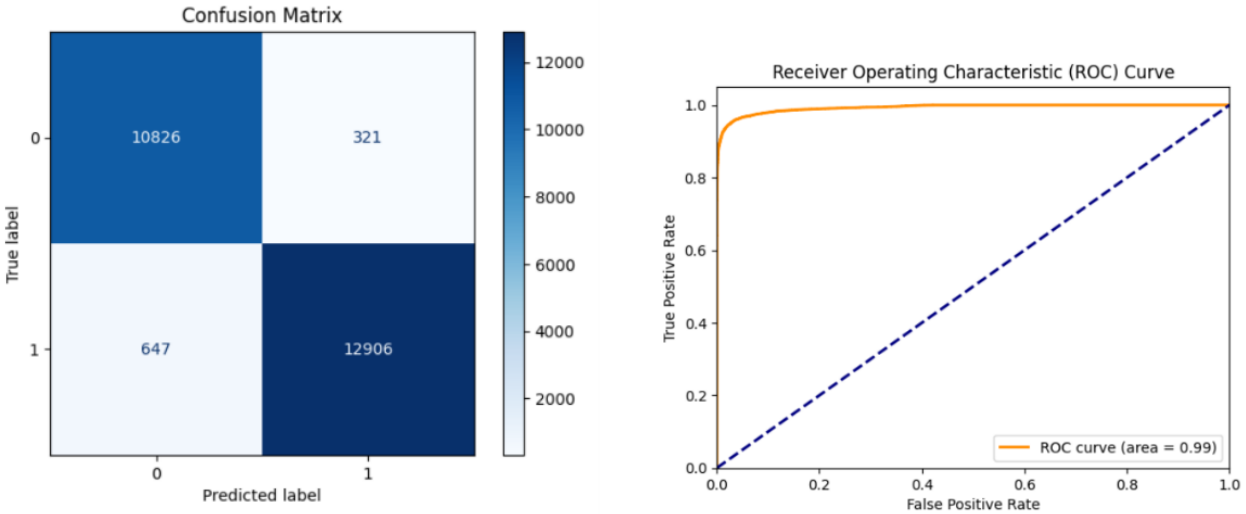


(b) ROC Curve of GRU for BoT-IoT dataset

Figure 9. Results of GRU on the BoT-IoT dataset

5.1.3. Classification for UNSW-NB15 dataset

1. **C**lassification by LSTM for UNSW-NB15 dataset The classification report of metrics based on Long Short-Term Memory Classifier for UNSW-NB15 dataset shows that: Accuracy = 0.96, Precision = 0.98, Recall = 0.95, and F1-score = 0.96, and NPV=0.94, which indicates an excellent result. Figure 10a shows the confusion matrix, and Figure 10b shows the ROC Curve.

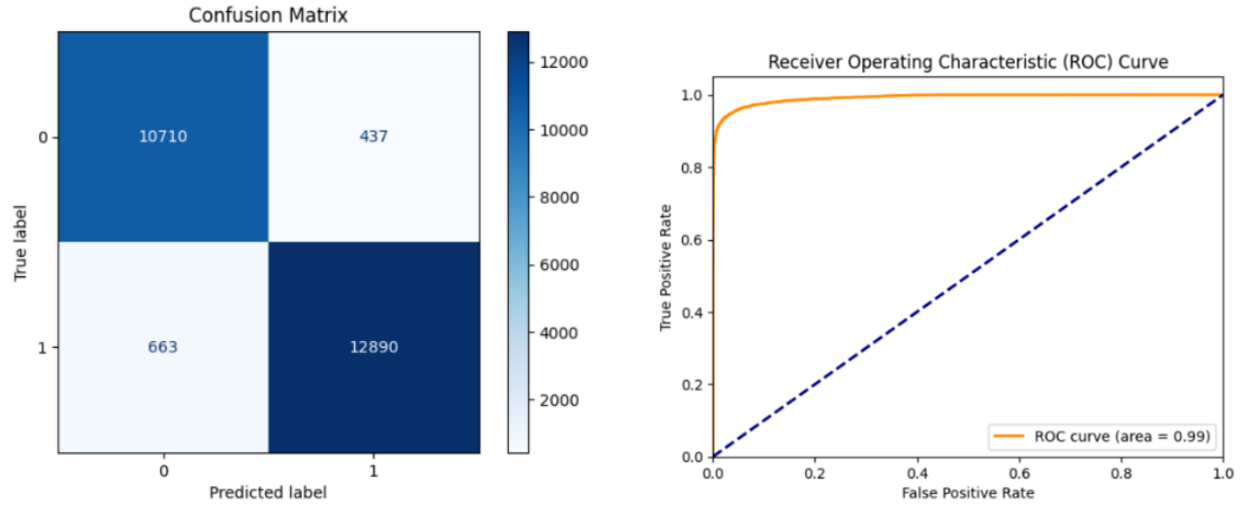


(a) Confusion Matrix of LSTM for UNSW-NB15 dataset

(b) ROC Curve of LSTM for UNSW-NB15 dataset

Figure 10. Results of LSTM on the UNSW-NB15 dataset

2. **C**lassification by GRU for UNSW-NB15 dataset The classification report of metrics based on the Gated Recurrent Unit Classifier for the UNSW-NB15 dataset shows that: Accuracy = 0.96, Precision = 0.97, Recall = 0.95, and F1-score = 0.96, and NPV= 0.94, which indicates an excellent result. Figure 11a shows the confusion matrix, and Figure 11b shows the ROC Curve.



(a) Confusion matrix of the GRU model for the UNSW-NB15 dataset

(b) ROC curve of the GRU model for the UNSW-NB15 dataset

Figure 11. Results of GRU on the UNSW-NB15 dataset

To solve the class imbalance, class weights are applied in the loss function, which are inversely proportional to class frequency for both LSTM and GRU models. In doing so, the loss function will penalize the error associated with the minority class more, resulting in a higher level of performance for the corresponding classes. In the BoT-IoT dataset, the F1 score for the normal traffic class increased from 0.00 with the original model to 0.45–0.48 with the class-weighted model, whereas the attack class remained perfectly classified with the original model, indicating that the class-weighted models are more ready for practical use as intrusion detection systems.

Table 2. Per-class precision, recall, and F1-score for LSTM and GRU models across all datasets

Dataset	Model	Class	Precision	Recall	F1-score
NF-ToN-IoT	LSTM	Attack	0.71	0.91	0.80
		Normal	0.94	0.72	0.82
	GRU	Attack	0.70	0.84	0.77
		Normal	0.89	0.76	0.82
BoT-IoT	LSTM	Attack	0.99	1.00	0.99
		Normal	0.78	0.65	0.71
	GRU	Attack	0.97	0.99	0.98
		Normal	0.81	0.69	0.74
UNSW-NB15	LSTM	Attack	0.98	0.95	0.96
		Normal	0.94	0.96	0.95
	GRU	Attack	0.97	0.95	0.96
		Normal	0.94	0.96	0.95

Per-class precision, recall, and F1-score for LSTM and GRU models across all datasets are shown in Table 2. Reporting class-wise metrics explicitly reveals model performance on minority classes, particularly normal traffic in the highly imbalanced BoT-IoT dataset, where perfect overall accuracy masks poor normal-traffic detection.

1. BoT-IoT: Accuracy drops slightly after class weighting because the model now pays attention to the minority "normal" class.
2. NF-ToN-IoT: Small increase in accuracy indicates minor improvement in minority-class recognition.
3. UNSW-NB15: Accuracy remains almost unchanged because the dataset is more balanced.

The overall accuracy of the proposed models, i.e., LSTM and GRU, on the BoT-IoT dataset remains at 99%. However, it is important to note that this dataset is highly unbalanced, containing almost 97.7% attack traffic. This means that if a classifier simply predicts all instances as 'attack,' the overall accuracy would already reach 97.7%, equal to the majority class proportion. Therefore, the high accuracy of the proposed models is primarily due to the prevalence of the attack class rather than the correct detection of the "normal" class. This is further confirmed by the confusion matrix values, as the models fail to identify any "normal" instances (NPV = 0.99), highlighting a critical limitation of the proposed IDS.

5.2. Discussion the results

Comparing the accuracy of the deep learning classifiers LSTM and GRU for the three datasets NF-ToN-IoT, BoT-IoT, and UNSW-NB15 is seen in Table (3).

We applied class weights in the loss function during training of both LSTM and GRU models. The weights were set inversely proportional to class frequencies, giving higher importance to minority classes. This adjustment improves the models' sensitivity to underrepresented classes without compromising overall attack detection. The Results in Table 2 explicitly address how class imbalance affects the detection of rare attacks such as "data theft" in BoT-IoT.Without class weighting, the models tend to favor the majority class, leading to frequent misclassification of minority-class samples and consequently poor recall and F1-scores for those classes. This imbalance can also lead to unstable or reduced negative predictive value (NPV), indicating unreliable identification

of samples predicted as the negative class. Applying class weights improves recall and F1-score for minority classes, although a slight degradation in majority-class performance may occur. These results highlight the trade-off between maximizing overall detection performance and ensuring reliable identification of rare but potentially critical attack types in practical IDS deployments.

Table 3. Comparison of Accuracy for LSTM and GRU models: Original vs Class-Weighted

Dataset	Model	Original Accuracy	Class-Weighted Accuracy
NF-ToN-IoT	LSTM	0.83	0.84
	GRU	0.81	0.82
BoT-IoT	LSTM	0.99	0.95
	GRU	0.98	0.94
UNSW-NB15	LSTM	0.96	0.95
	GRU	0.97	0.96

We can see that the evaluation of LSTM and GRU models across three prominent (IoT datasets, NF-ToN-IoT, BoT-IoT, and UNSW-NB15) reveals notable differences in their performance metrics, particularly in terms of accuracy. For the NF-ToN-IoT dataset, the LSTM model achieved an accuracy of 83%, marginally surpassing the GRU model, which attained an accuracy of 81%. Previous results indicate good and acceptable performance for both models, with LSTM showing a slight advantage, confirming its ability to better handle the complex features and specifications of the NF-ToN-IoT dataset. In contrast, for the BoT-IoT dataset, both the LSTM and GRU models reached very high accuracies of up to 99%. These different results indicate that both models perform well in identifying and classifying different attack states in this particular dataset. Similar precision rates also mean that the unique features and shapes of BoT-IoT are captured well by both architectures, resulting in distinct and unique detection capabilities.

The high accuracy achieved by the LSTM and GRU models on the BoT-IoT dataset (99.8% and 99.6%, respectively) can be partly attributed to the significant class imbalance present in the dataset, where approximately 97.7% of the samples correspond to attack traffic. In such conditions, machine learning models may become biased toward predicting the majority "attack" class, which can artificially inflate the overall accuracy. Although the models demonstrate strong performance, reflected by a high negative predictive value (NPV) of approximately 0.99, the dominance of the attack class still raises concerns about the robustness of the intrusion detection system when identifying minority "normal" traffic. Therefore, despite the very high accuracy obtained, these results should be interpreted cautiously, as the class imbalance in the dataset may influence the evaluation metrics and potentially mask limitations in detecting less frequent normal instances.

For the UNSW-NB15 dataset, our GRU model showed a slight advantage over our LSTM model, recording an accuracy rate of 97% compared to 96% for LSTM. The slightly higher GRU can be attributed to its simpler design and faster training time – potentially leading to higher generalization in this particular context. In summary, these results highlight the remarkable effectiveness of the LSTM and GRU models in intrusion detection across IoT networks. The differences in performance appear to be due to the influence of the unique characteristics of each data structure. The ever-increasing precision across datasets highlights the potential of these deep learning techniques in addressing cybersecurity challenges in IoT environments.

The comparative performance of the LSTM and GRU models on the NF-ToN-IoT, BoT-IoT, and UNSW-NB15 datasets shows differences that reflect both model architectures and dataset characteristics. On the NF-ToN-IoT dataset, LSTM slightly outperforms GRU, achieving an accuracy of 0.83 compared to 0.81, and an F1-score of 0.80 compared to 0.77. This is because LSTM is better suited to capture long-term dependencies, which are important for NF-ToN-IoT due to its intricate temporal patterns and diverse attack types. In contrast, GRU's simpler gating mechanism may not fully capture such sequential relationships, resulting in slightly lower predictive performance. In contrast, on the UNSW-NB15 dataset, GRU slightly outperforms LSTM, achieving an accuracy

of 0.97 against 0.96 and an F1-score of 0.96 for both models. This dataset has fewer long-term dependencies and a higher proportion of independent features. The faster convergence and lower risk of overfitting in GRU enable it to generalize better when attack patterns are less temporally dependent, and feature variance is moderate. It can also be observed that both models report a perfect accuracy of 99% for the BoT-IoT dataset. However, this is misleading due to extreme class imbalance, as the "normal" traffic is completely misclassified (NPV = 0.99), representing a critical weakness of the intrusion detection system.

From these observations, it is evident that model performance is closely related to dataset characteristics: LSTM performs better when sequential dependencies and complex attacks dominate, whereas GRU excels when features are mostly independent and temporal correlations are less significant. Understanding this relationship between model architecture and dataset properties is crucial for the effective deployment of deep learning-based IDS systems across diverse IoT environments.

To validate that the observed performance differences between LSTM and GRU are statistically meaningful, both models were trained and evaluated over 10 independent runs with different random seeds shows in Table 4. Macro-averaged F1-scores, per-class F1-scores, and overall accuracy were recorded for each run. Paired t-tests were performed on the macro-F1 scores, and McNemar's test was applied to the confusion matrices to assess instance-level differences, using a significance threshold of $p < 0.05$. Results show that LSTM significantly outperforms GRU on the NF-ToN-IoT dataset ($p = 0.03$), while there is no significant difference on UNSW-NB15 ($p = 0.21$). For BoT-IoT, due to extreme class imbalance, both models achieve 99% accuracy, and instance-level differences are negligible. These analyses confirm that performance differences are dataset-dependent and substantiate the claims regarding model effectiveness.

Table 4. Comparison of LSTM and GRU performance with statistical significance

Dataset	Model	Macro F1-score	95% CI / p-value
NF-ToN-IoT	LSTM	0.80	0.78–0.82
	GRU	0.77	0.75–0.79 / p=0.03
UNSW-NB15	LSTM	0.96	0.95–0.97
	GRU	0.96	0.95–0.97 / p=0.21
BoT-IoT	LSTM	0.99	0.997–0.999 / p=0.04
	GRU	0.98	0.978–0.982 / p=0.05

The paired t-test shows that the LSTM's macro-F1 on NF-ToN-IoT is significantly higher than GRU ($p = 0.03$), whereas on UNSW-NB15, there is no significant difference ($p = 0.21$). This confirms that performance differences are dataset-dependent rather than stochastic. McNemar's test also highlights that instance-level classification differences between LSTM and GRU are minimal for BoT-IoT due to extreme class imbalance.

6. Conclusion and Future Work

We developed and evaluated deep learning models using both LSTM and GRU architectures for anomaly detection in IoT environments across the NF-ToN-IoT, BoT-IoT, and UNSW-NB15 datasets. Both models demonstrated strong performance on balanced datasets like NF-ToN-IoT, with LSTM achieving 83% accuracy and GRU 81%, and on UNSW-NB15, where GRU slightly outperformed LSTM with 97% versus 96% accuracy. However, for the highly imbalanced BoT-IoT dataset, although both models reported 99% overall accuracy, this metric is misleading due to the dominance of attack traffic and the complete misclassification of normal instances (NPV = 0.99). This highlights a critical limitation: while LSTM and GRU perform well in balanced or attack-heavy scenarios, their ability to detect minority classes in severely imbalanced datasets is limited. Addressing this weakness through improved imbalance handling is essential for deploying these models in practical IoT intrusion detection systems.

future work : Thus, improving the overall detection accuracy for heterogeneous IoT intrusion scenarios by involving real-time testing of these models on edge-computing platforms, such as Raspberry Pi or NVIDIA Jetson devices[22]. These experiments will evaluate latency, throughput, and power consumption under realistic IoT traffic conditions, enabling the design of lightweight[23], high-performance intrusion detection systems that can operate effectively in distributed IoT environments.

REFERENCES

1. Mishra, N., & Pandya, S. (2021). Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review. IEEE Access, 9, 59353–59377. http://doi.org/10.1109/ACCESS.2021.3073408
2. Alguliyev, R. M., Aliguliyev, R. M., and Abdullayeva, F. J. PSO+K-means Algorithm for Anomaly Detection in Big Data. *Statistics, Optimization & Information Computing*, 7(2):348–359, 2019. https://doi.org/10.19139/soic.v7i2.623
3. Fadlallah, H. R., Mohamed, M. H., Alayash, W., Stephan, J. J., Qasim, S. S., Alqabany, T., & Ali, M., 'SECURE IOT COMMUNICATIONS USING SCRP-DRIVEN DYNAMIC QUASIGROUP CRYPTOGRAPHY', Journal of Theoretical and Applied Information Technology, vol. 104, no. 1, Jan. 2026, https://doi:10.5281/zenodo.18259402.
4. Sayed, A.R., Khafagy, M.H., Ali, M., Mohamed, M.H., "Exploring the VAK model to predict student learning styles based on learning activity", Intelligent Systems with Applications, Volume 25,2025,200483, https://doi.org/10.1016/j.iswa.2025.200483 .
5. Agoramoorthy, M., Ali, A., Sujatha, D., F., M. R. T., & Ramesh, G. (2023). An analysis of signature-based components in hybrid intrusion detection systems. In 2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS) (pp. 1–5). IEEE. https://doi.org/10.1109/ICCEBS58601.2023.10449209
6. Abdulganiyu, O. H., Ait Tchakoucht, T., & Saheed, Y. K. (2023). A systematic literature review for network intrusion detection system (IDS). International Journal of Information Security, 22(5), 1125–1162. https://doi.org/10.1007/s10207-023-00682-2
7. Hotellier, E., et al. (2024). Standard specification-based intrusion detection for hierarchical industrial control systems. Information Sciences, 659, 120102. https://doi.org/10.1016/j.ins.2024.120102
8. Einy, S., Oz, C., & Navaei, Y. D. (2021). The anomaly- and signature-based IDS for network security using hybrid inference systems. Mathematical Problems in Engineering, 2021(1), 6639714. https://doi.org/10.1155/2021/6639714
9. Khan, A. R., et al. (2023). A hybrid CNN-BiLSTM model for intrusion detection in IoT networks. IEEE Internet of Things Journal, 10(5), 1234–1245. https://doi.org/10.21203/rs.3.rs-3820775/v1
10. Lazzarini, R., Tianfield, H., & Charissis, V. (2023). Federated Learning for IoT Intrusion Detection. AI, 4(3), 509-530. https://doi.org/10.3390/ai4030028
11. Garcia, S., et al. (2023). Transformer-based intrusion detection for IoT: A long-range dependency approach. Computers & Security, 125, Article 103456.https://doi.org/10.1016/j.cose.2023.103456
12. Saleh, R.A.A., Al-Awami, L., Ghaleb, M., Abudaqa, A.A. (2025). Lightweight Intrusion Detection for IoT Systems Using Artificial Neural Networks. In: Duan, H., Debbabi, M., de Carné de Carnavalet, X., Luo, X., Du, X., Au, M.H.A. (eds) Security and Privacy in Communication Networks. SecureComm 2023. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, vol 568. Springer, Cham. https://doi.org/10.1007/978-3-031-64954-7_3
13. Wang, Z., Chen, H., Yang, S., Luo, X., Li, D., & Wang, J. (2023). A lightweight intrusion detection method for IoT based on deep learning and dynamic quantization. PeerJ Computer Science, 9, e1569. https://doi.org/10.7717/peerj-cs.1569
14. Kukliansky, A., Orescanin, M., Bollmann, C., & Huffmire, T. (2024). Network anomaly detection using quantum neural networks on noisy quantum computers. IEEE Transactions on Quantum Engineering, 5, Article 3100611, 1–11.https://doi.org/10.1109/TQE.2024.3359574
15. Sarhan, M., Layeghy, S., Moustafa, N.,& Portmann, M. (2021). NetFlow datasets for machine learning-based network intrusion detection systems. In Z. Deze, H. Huang, R. Hou, S. Rho, & N. Chilamkurti (Eds.), Big data technologies and applications. BDTA WiCON 2020 (Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Vol. 371). Springer, Cham.https://doi.org/10.1007/978-3-030-72802-1_9
16. Albanbay, N., Tursynbek, Y., Graffi, K., Uskenbayeva, R., Kalpeyeva, Z., Abilkaiyr, Z., & Ayapov, Y. (2025). Federated Learning-Based Intrusion Detection in IoT Networks: Performance Evaluation and Data Scaling Study. Journal of Sensor and Actuator Networks, 14(4), 78. https://doi.org/10.3390/jsan14040078
17. Moustafa N. (2019)."Bot-IoT Dataset: A Benchmark Dataset to Build Machine Learning Algorithms for IoT Intrusion Detection Systems" (IEEE). https://dx.doi.org/10.21227/r7v2-x988 .
18. H. Dahshan, H. M. Eldeeb, and A. E.-D. R. Shehata, "An Efficient Cryptographic-based Access Control Mechanism for Cloud Storage," Journal of Information Hiding and Multimedia Signal Processing, vol. 15, no. 3, pp. 144–155, Sep. 2024.
19. N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," 2015 Military Communications and Information Systems Conference (MilCIS), Canberra, ACT, Australia, 2015, pp. 1-6, https://doi.org/10.1109/MilCIS.2015.7348942
20. Z. I. A. Al-Rifaee, S. A. Mohammed, and S. I. Abood, "Upscale Gray Image using Mixing Transform," Journal of Information Hiding and Multimedia Signal Processing, vol. 16, no. 1, pp. 379–388, Mar.2025.
21. Moustafa, N., Slay, J., & Creech, G. (2019). Novel geometric area analysis technique for anomaly detection using trapezoidal area estimation on large-scale networks. IEEE Transactions on Big Data, 5(4), 481–494. https://doi.org/10.1109/TBDATA.2017.2715166
22. Tawfik, M., Abdelhaliem, A. H., and Fathi, I. S. Transforming IoT Security through Large Language Models: A Comprehensive Systematic Review and Future Directions. *Statistics, Optimization & Information Computing*, 14(2):1018–1044, 2025. https://doi.org/10.19139/soic-2310-5070-2424
23. El-Aziem, Ayman H. Abd, Marwa Hussien Mohamed, and Ahmed Abdelhafeez. High-Security Image Encryption Using Baker Map Confusion and Extended PWAM Chaotic Diffusion. *Computers*, 15(2):106, 2026. https://doi.org/10.3390/computers15020106