AN ADAPTIVE SPATIOTEMPORAL DEFENSE FRAMEWORK FOR ROBUST AND EFFICIENT INTRUSION DETECTION IN IOT NETWORKS

Zarinabegam K. Mundargi[1,2*], ***Azra Nasreen***[1]

[1]Department of Computer Science and Engineering, R.V College of Engineering, Visvesvaraya Technological University, Belagavi, Karnataka. India, 590018

[2]Department of Artificial Intelligence and Data Science, Vishwakarma Institute of Technology, Pune, Maharashtra, India, 411037

Emails: zarin1100@gmail.com[1,2*] (Corresponding author), azranasreen@rvce.edu.in[1]

Abstract

The ever-increasing growth rate of the Internet of Things (IoT) has significantly increased network heterogeneity and data volumes, which have posed a significant risk of various cyber threats to resource-constrained IoT devices. Traditional Intrusion Detection Systems (IDS) may not perform well in such environments, which may be characterized by noisy data, class imbalance, and a large number of feature redundancies. This study proposes an Adaptive Spatiotemporal Defense Framework suitable for real-world IoT networks. The framework includes an efficient nonlinear data preprocessing approach, discriminative feature optimization, and a spatiotemporal learning framework to improve the performance of intrusion detection systems. The framework efficiently addresses the class imbalance problem, reduces redundant features, and maintains discriminative features, thereby improving the quality of the features and reducing the complexity. The inclusion of spatiotemporal learning in the framework enables the detection of complex data relationships with high precision. The performance of the proposed approach is evaluated on three benchmarking datasets: UNSW-NB15, ToN-IoT, and IoT-23. The accuracy obtained on these datasets is 89.61%, 97.94%, and 99.99%, respectively. Along with this, high F1-score results are obtained. This proves the enhancement in intrusion detection ability. Overall, the proposed framework proves to be a reliable, efficient, and lightweight solution for intrusion detection in heterogeneous IoT environments.

Keywords: ***IoT Security; Intrusion Detection System; Spatiotemporal Learning; Feature Optimization; Pattern Preservation; Adaptive Framework; Anomaly Detection.***

1.0 INTRODUCTION

The rapid growth of the Internet of Things has led to the widespread deployment of heterogeneous resource-constrained devices in critical application domains such as healthcare, industrial automation, smart homes, and transportation, thereby significantly enhancing the attack surface of modern networks [1-4]. Despite their practical advantages, IoT networks are naturally prone to security threats because of their limited processing, memory, and energy resources, which restrict the application of traditional security countermeasures [5-7]. Intrusion Detection Systems have, therefore, been identified as a critical security countermeasure to detect potential threats by monitoring network activities. However, traditional Intrusion Detection Systems are ineffective for IoT networks because they fail to handle noisy data, feature dimensionality, as well as class imbalance issues, thereby compromising detection accuracy and resulting in higher false alarms [8, 9].

To overcome these challenges, artificial intelligence (AI)-based methods, particularly machine learning (ML) and deep learning (DL), have been extensively studied in the context of IoT intrusion detection systems. ML-based methods are found to be effective in providing automated pattern recognition capabilities, but are not suitable for modeling complex relationships in large-scale dynamic IoT communications while being resource-efficient [10]. Contrarily, DL methods, including CNN, RNN, LSTM, etc., are found to provide better feature extraction capabilities in modeling complex relationships in large-scale dynamic communications while providing better intrusion detection capabilities in identifying sophisticated and dynamic cyberattacks in IoT communications [11,12]. However,

deploying DL-based IDS in large-scale dynamic IoT communications is difficult due to its computational complexities while requiring high training and low adaptability in modeling dynamic and heterogeneous cyberattacks in IoT communications [13]. Moreover, adversarial ML and data poisoning attacks are found to compromise the reliability and robustness of DL-based IDS in IoT communications [14].

Another significant limitation of existing methods lies in their failure to handle various problems simultaneously, such as the presence of redundant features, the existence of an unbalanced distribution of attacks, and the maintenance of significant spatiotemporal correlations within network traffic. Existing methods have mostly adopted static preprocessing techniques or simple feature selection methods, which may cause the loss of significant features or may lead to biased learning towards specific classes. Also, existing methods have mostly adopted overall evaluation mechanisms, with less emphasis on specific classes of attacks and deployment constraints, which may limit their overall generalization capabilities for real-world IoT scenarios.

With all these limitations, this study proposes a novel framework for intrusion detection that is not only efficient in terms of resource consumption within the IoT environment but also provides higher detection capabilities. For this purpose, the proposed framework uses advanced techniques for data preprocessing, feature optimization, and deep learning to provide accurate detection capabilities. Specifically, the proposed framework is designed to address the limitations of existing IDS methodologies that are either not efficient in terms of resource consumption within the IoT environment or provide accurate detection capabilities. With these limitations in mind, the proposed framework will contribute to the development of more efficient intrusion detection solutions. The major contributions of this research can be summarized as follows:

1) This research aims to provide a strong data preprocessing method that will handle the issue of class imbalance without losing the nonlinear relationships between the data attributes. This will improve the sensitivity of the detection system, especially for rare but impactful types of attacks.
2) A novel method of feature refinement is developed, which focuses only on the most important attributes of the data. This will eliminate the presence of redundant attributes, which will improve the efficiency of the detection system.
3) A deep learning method is proposed, which considers the spatial as well as the temporal aspects of the data. This will improve the detection system's ability to differentiate between normal and malicious traffic in dynamic environments.
4) Moreover, the framework is designed to efficiently handle the limited resources of IoT systems. It provides a balance between efficiency and accuracy, which will allow it to be used across different IoT environments.

The remainder of this manuscript is organized as follows. Section 2.0 presents a comprehensive review of existing literature, highlighting key research gaps and limitations. Section 3.0 describes the proposed methodology in detail. Section 4.0 outlines the experimental setup and presents the performance evaluation of the proposed framework across multiple benchmark datasets. Section 5.0 provides the results and corresponding analysis, followed by Section 6.0, which discusses the findings in depth. Finally, Section 7.0 concludes the manuscript by summarizing the key contributions and suggesting potential directions for future research.

2.0 LITERATURE REVIEW

Recent research has reported a significant increase in the utilization of deep learning and hybrid learning methods for the detection of IoT-based intrusions in response to increasingly sophisticated and ever-changing cyber threats in the form of cyber-attacks. These methods use the capabilities of deep learning to provide higher accuracy in detecting attacks by characterizing non-linear traffic flows and their temporal dependencies. Despite the advantages of these hybrid and deep learning methods, there are still many limitations that have yet to be addressed, including scalability to a large number of IoT devices, resolving the class imbalance issue, and generalizing to unique or rare types of attack vectors.

Popoola et al. [15] developed a hybrid Bi-LSTM-based solution combined with an autoencoder for dimensionality reduction of attack vector data and botnet detection. This solution performed well on the BoT-IoT dataset, but because of its reliance on a limited number of datasets and its aggressive compression of the attack vector features, the identification of minority attack patterns may have been hindered. Similarly, Zeeshan et al. [16] proposed a protocol-based DDOS detection architecture designed to identify DDOS and DoS intrusion attacks. This approach does have advantages over many existing protocols; however, its scope is limited to the types of attacks/ domains it was designed for, and it may not be able to identify sophisticated or emerging types of attacks.

Table 1: Overview of literature review

Author(s)	Methodology	Key Features	Limitations
Popoola et al. [15]	A Hybrid Deep Learning approach combining BLSTM and dimensionality reduction via Autoencoder.	Effectively analyzes temporal variations in IoT traffic and reduces feature dimensionality for efficient processing.	Reliance on a single dataset may lead to potential blind spots in detecting diverse attack types due to dimensionality reduction.
Zeeshan et al. [16]	PB-DID architecture using DL for classifying normal, DoS, and DDoS traffic.	A protocol-oriented learning and classification approach designed for IoT-specific traffic patterns.	May not detect novel or complex attacks (e.g., APTs or data breaches); high computational cost in large-scale deployments.
Zhao et al. [17]	Lightweight Deep Neural Network (LNN)-based NID using PCA and efficient architectural structures.	Incorporates channel shuffle, inverse residual, and expansion–compression layers for reduced model size and improved speed.	Possible information loss during dimensionality reduction; limited adaptability to emerging or unknown attacks.
Sharma et al. [18]	Anomaly-based IDS using DNN with filter-based feature selection and GAN-generated synthetic samples.	Combines deep feature selection and data augmentation to improve recognition of minority attack types.	Limited performance on real data; GAN effectiveness depends on synthetic data quality and attack diversity.
Moualla et al. [19]	IDS using Extremely Randomized Trees (Extra Trees Classifier) and SMOTE for data balancing.	Dynamic, scalable framework with improved precision-recall and low false alarm rates.	Imbalanced attack distributions in the dataset may bias detection toward dominant attack classes.
Alsharaiah et al. [20]	Network IDS based on LSTM integrated with attention mechanisms.	Captures temporal dependencies and key feature relevance using attention-enhanced sequence modeling.	Prone to overfitting due to high model complexity and risk of memorizing training data rather than generalizing.
Akif et al. [21]	Hybrid voting-based IDS integrating RF, XGBoost, KNN, and AdaBoost for IoT intrusion detection.	• Improved detection accuracy through ensemble learning • Performs well in both binary and multi-class classification • Validated on IoT-23 dataset • Handles evolving IoT threats effectively	• Latency issues in real-time environments • High computational cost • Less suitable for resource-constrained IoT devices

To reduce the computational burden of IoT, some researchers have attempted to use lightweight Neural Networks (LNNs) to detect intrusions more efficiently (Zhao et al. [17]). The authors of [17] demonstrated how dimension-reduction techniques used to develop LNN-based ID systems have improved efficiency by reducing the number of features to be monitored; however, some of these techniques also risk losing discriminative features that may be necessary for the detection of minority classes. Sharma et al. [18] have proposed combining DBB with augmentation methods to address issues related to class imbalance; however, the success of augmenting training data used to create synthetic training samples is dependent on both the quality of the training data and the stability of the model used to train the data.

Both tree-based approaches and ensemble-based approaches (utilizing a classifier based on SMOTE-enhanced classifiers, Moualla et al. [19], and a hybrid voting system, Akif et al. [21]) have been shown to improve accuracy for class imbalance. However, these approaches can introduce high latency and scalability issues, making them unsuitable for deployment on IoT-edge devices with limited resources. Attention-based LSTM model Alsharaiah et al. [20] effectively capture temporal dependencies within the input data; however, they tend to overfit data, limiting their ability to generalize and provide meaningful results with respect to per-class failure cases. The overall finding of the reviewed literature is indicated in Table 1.

In general, most of the existing IoT intrusion detection research has focused on evaluating performance in aggregate terms, rather than providing a detailed analysis of performance degradation associated with specific classes, particularly rare attack classes. As a consequence, many of the proposed techniques suffer from either high computational complexity or loss of information associated with feature reduction. Thus, there is an opportunity to create a cohesive, lightweight, and dynamic ID system that can accommodate class imbalance, feature relevance, and the incorporation of both spatial and temporal elements to better address these limitations.

3.0 METHOD

The rapid expansion of IoT Deployments for smart environments, industrial automation, and healthcare has created new opportunities as well as increased security risk. The fact that IoT has resulted in numerous disparate traffic patterns, limited available computing resources, and continually evolving cyber threats has resulted in greater complexity when trying to create and implement IDS that are reliable and scalable. Existing IDS solutions are only intended to provide baseline production and generally do not detect advanced or emerging attack behaviors, exhibit inconsistent data quality, and do not meet real-time performance requirements. In addition, many of the current state-of the art IDS architectures utilize a large number of computing-intensive components that require substantial amounts of memory and processing power, making them unsuitable for the large-scale and resource-constrained nature of IoT environments. To overcome these limits, the proposed Adaptive Spatiotemporal Defense framework provides a unified multi-stage architecture for preprocessing, feature optimization, and Spatiotemporal deep learning to allow reliable IoT ID. Insufficient understanding of the current IDS solution obstacles for scaling and accuracy. One of the major problems is the loss of critical information during dimensionality reduction, which decreases classification performance and limits the system's ability to detect critical or novel attack patterns. The proposed framework suggests minimizing computational overhead while enabling better adaptability and detection accuracy. Another limiting factor is that the underlying class imbalance of IoT intrusion datasets severely underrepresents the minority of attack categories, resulting in an unequal bias in learning, leading to lower sensitivity for identifying rare but high-risk threats. To provide solutions for these limitations, an Adaptive Nonlinear Data Processing Module (ANDPM) was developed. ANDPM is a hybrid preprocessing mechanism using a support vector machine (SVM) to reconstruct nonlinear patterns from incoming traffic and apply synthetic sampling to create meaningful instances of minority attack categories (using an ADASYN-driven mechanism). The overall effect of ANDPM on the data distribution is to create a more balanced and representative dataset, enabling the model to generalize across many intrusion scenarios encountered in the IoT environment. An additional challenge for IoT environments is the high dimensionality and volatility of traffic patterns. The high degree of dimensionality creates additional computational overhead that reduces the learning efficiency of machine learning models and increases the risk of deleting essential discriminative information. The solution to this limitation is to introduce Adaptive Discriminative Feature Optimization (ADFO), which dynamically refines features identified by the model. Drawing upon a combination of goal-directed and coordinated search strategies, ADFO identifies and retains only the most relevant attributes, eliminating any that may be redundant. The effect of ADFO is to reduce the complexity of the model and provide a mechanism for improving learning efficiency while preserving critical, discriminative patterns. To correctly detect intrusions, an accurate detection system must be able to model both spatial structures and temporal dependencies of IoT traffic flows. However, conventional models often struggle to accurately model these structures. The Deep SpatioTemporal Fusion

Network (DSTF-Net) has been developed to address these flaws in traditional intrusion detection techniques. The convolutional layers within the DSTF-Net capture spatial variability within the network traffic, while the bidirectional LSTM component captures temporal variability in both directions (forward and backward). This complementary architecture allows the DSTF-Net to identify many different, complex, evolving, and multi-stage intrusions. By employing ANDPM for robust data preprocessing, ADFO for optimized feature refinement, and DSTF-Net for enhanced spatio-temporal learning, the methodology presented is capable of successfully addressing the problems of inefficiency related to computational resources, class imbalance, high dimensionality, and loss of significant information. All of these components, when combined into a single framework for intrusion detection, can create a very accurate, scalable, and adaptable system for use in today's changing, resource-constrained IoT environments. A block diagram illustrating the entire proposed approach is provided in Fig. 1.

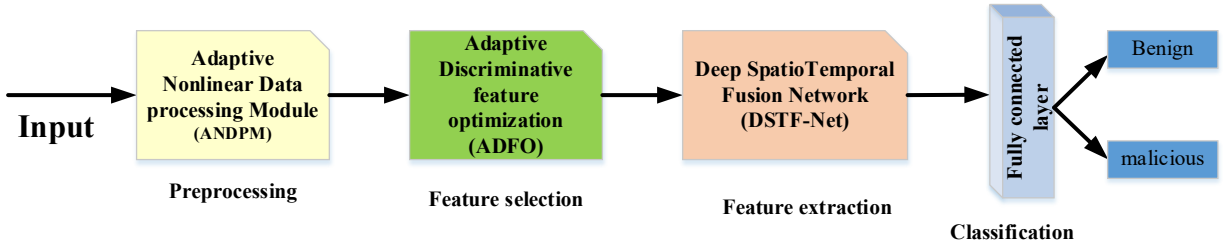

Fig. 1: Block diagram illustrating the proposed approach

3.1 Adaptive Nonlinear Data Processing Module (ANDPM)

Preprocessing is an important step in deciding the accuracy, reliability, and ability to generalize an intrusion detection system. In an IoT environment with high-dimensional data, existing data preprocessing methods are not able to maintain complex nonlinear relationships among features while dealing with data imbalance issues. Oversampling methods are not able to produce quality oversampled data because they introduce noisy data into the data set. This can cause overfitting in an intrusion detection system. Therefore, an adaptive data preprocessing strategy is required to maintain data quality while balancing classes.

To address these issues, the proposed Adaptive Nonlinear Data Processing Module (ANDPM) unifies Support Vector Machine-based imputation and Adaptive Synthetic Sampling (ADASYN) in a single module. The Support Vector Machine-based imputation module uses the principles of Support Vector Regression to address missing values in the data, thus maintaining non-linear relationships between the features. Unlike traditional statistical imputation techniques, this module preserves the underlying distribution in the data, thus enhancing the quality of the input to the next learning stages. Once imputation is complete, the ADASYN approach is used, which balances class distribution by appropriately synthesizing minority class instances. Rather than equally synthesizing minority class instances, as in the case of oversampling, the ADASYN approach focuses on regions of the data space that are sparsely populated or harder to learn, thus providing a more sensitive classifier against rare attack patterns. This approach prevents the problem of duplication and noise, which is a drawback of oversampling methods like SMOTE.

In addition to these basic components, feature-wise normalization is used in ANDPM to normalize the scale of the input variables. The normalization ensures that the input features contribute equally to the model's training. The categorical features are transformed appropriately to preserve their statistical properties. Additionally, a hybrid validation is conducted on synthetic data, where distance and variance thresholds are employed to filter out any unrealistic synthetic data. The synthetic data is expected to have meaningful feature distributions. The final result of the ANDPM process is a well-balanced dataset that is refined to capture nonlinear relationships between features. This process not only removes noise and redundancy but also enhances the quality of the data. This ensures that learning becomes more effective at later stages of the proposed framework. As a result, this module helps to achieve improved detection accuracy, as well as robustness to various challenges that are usually encountered during intrusion detection, especially within IoT environments.

3.2 Adaptive Discriminative Feature Optimization (ADFO)

Feature selection is considered to be very important in enhancing the efficiency and efficacy of intrusion detection systems. This is particularly true in the case of IoT networks, which tend to have high-dimensional and heterogeneous data. Irrelevant features tend to increase computational complexity and reduce the efficacy of classification algorithms by introducing "noise" in the features. Keeping these challenges in view, this paper proposes a novel feature selection technique called Adaptive Discriminative Feature Optimization (ADFO), which utilizes a "metaheuristic" optimization technique based on the Artificial Fish Swarm Algorithm (AFSA).

In ADFO, every candidate solution is a set of features represented as a binary number, where features are represented as "1" if they are selected or as "0" if they are not. The optimization process involves iteratively improving these sets of features based on a fitness function that assesses their classification performance. The search process is controlled by three basic behaviors: follow, swarm, and prey. These three behaviors work together to allow exploration and exploitation of the search space. The "follow" behavior enables candidate solutions to be attracted to better-performing candidate solutions, the "swarm" behavior causes convergence to promising regions, and the "prey" behavior adds randomness to prevent convergence to local optima. These steps are performed iteratively until a termination criterion is met. In order to promote adaptability, the ADFO method uses a dynamic search strategy that varies parameters such as neighborhood range and movement strategy according to the fitness distribution of the population. This process helps to promote convergence stability while maintaining diversity within the search space. The evaluation of fitness mainly considers the relevance of the selected features to the accuracy of the classifier. This ensures that the selected subset of features does not compromise the accuracy of the classifier. Other constraints are used to regulate interactions between the solutions.

The interaction between candidate solutions in ADFO is controlled by several mathematical formulations. The distance function (Equation (1)) measures the similarity between feature subsets, which is used in the formation of neighborhoods.

$$dist(Fe_i, Fe_j) = \sum_{k=1}^{k} \left| Fe_i(k) - Fe_j(k) \right| \tag{1}$$

In the distance function, Fe_i and Fe_j represent two candidate feature subsets, while k denotes the total number of features. The dynamic vision mechanism incorporates fitness-related terms, where Ff_i is the fitness of the current solution, Ff_{max} and Ff_{avg} represent the maximum and average fitness of the population, respectively, and Ff_{i-1},Ff_{i+1} indicate neighboring fitness values. The functions F_1 and F_2 are adaptive scaling factors, and U is a random variable controlling stochastic search behavior.

The dynamic vision mechanism (Equation (2-4)) is used to control the range of the search. The balance between global search and local exploitation is ensured by the adaptive adjustment of the range.

$$\mathfrak{E}\delta(i) = \mathfrak{F}\mathfrak{f}_1\left(\frac{\mathfrak{F}\mathfrak{f}_{max} - \mathfrak{F}\mathfrak{f}_i}{\mathfrak{F}\mathfrak{f}_{max} - \mathfrak{F}\mathfrak{f}_{avg}}\right) \cdot \left[\frac{\pi}{2} + \mathfrak{F}\mathfrak{f}_2\left(\mathfrak{F}\mathfrak{f}_i - \frac{\mathfrak{F}\mathfrak{f}_{i-1} + \mathfrak{F}\mathfrak{f}_{i+1}}{2}\right)\right] \tag{2}$$

$$Dynamic\ Vision(i) = Vision\ (Fe_i) \cdot \mathfrak{E}\delta(i) \cdot \mathfrak{U} \tag{3}$$

$$Vision\ (Fe_i) = \frac{\sum_{k=1}^{k} \sum_{k=1}^{N} |Fe_i(k) - Fe_j(k)|}{Total\ number\ of\ fishes} \tag{4}$$

The search strategy (Equation (5)) is used in the search for candidate solutions in the effective search space.

$$Neig(Fe_i) = \{Fe_k | 0 < dist(Fe_i, Fe_k) \leq dynamic\ vision\} \tag{5}$$

The detection of the center (Equation (6)) of a set of candidate solutions is also ensured.

$$Cent(Fe_i) = Fe_{center}(i) = \begin{cases} 0 & \sum_{k=1}^{k} Fe_k(i) < \frac{k}{2} \\ 1 & \sum_{k=1}^{k} Fe_k(i) \geq \frac{k}{2} \end{cases} \tag{6}$$

The density of the population, which prevents excessive convergence in a specific region, is ensured by the crowd degree (Equation (7)). The optimization is ensured by the fitness function (Equation (8)), which evaluates the quality of the feature subsets.

$$\mathfrak{ED}(Fe_i) = \frac{Neighbors\ of\ Fe_i}{Total\ number\ of\ fishes} \tag{7}$$

$$F = b \times (y_{ji}^2 - y_{js})^2 + (a - y_{ji})^2 \tag{8}$$

In the vision computation, N represents the total number of candidate solutions (fishes), while the expression represents the average distance between the candidate solutions. The neighbor set $Neig(Fe_i)$ represents the candidate solution set within the adaptive vision range, which allows local search. The center detection function $Cent(Fe_i)$ is used to find the representative position of neighboring candidate solutions, which is achieved by the majority selection of the feature set. The crowd degree $\mathfrak{ED}(Fe_i)$ It is used to avoid clustering, which is important to increase diversity. The fitness function is used to evaluate the quality of the feature subset, where a, b, y_{ji}, and y_{js} The model parameters related to the classification error and output targets. The above parameters guarantee the balance between exploration and exploitation, which allows the swarm to converge to the optimal solution stably.

The overall process of the feature selection process can be represented by the following algorithm, denoted as Algorithm 1 - ADFO for Feature Selection. As represented by this algorithm, the overall process of feature selection involves the process of initialization of feature subsets, optimization of features by the application of behavioral operators, and the selection of the optimal feature subset based on fitness evaluation. Thus, the overall process of feature selection by the application of ADFO helps to simplify the complexity of the learning process by eliminating redundant features.

Algorithm 1: ADFO for Feature Selection

Initialize the fish swarm or features.
For the next fish i, where $i \leq N$
  Evaluate the best fish swarm position based on the current feature set.
  Compute the dynamic vision (Fe_i) for fish i
  Do Follow Step
    If the following step succeeds:
      Proceed to the next fish.
    Otherwise:
  Do Swarm Step:
    If the swarm step succeeds:
      Proceed to the next fish.
    Otherwise:
  Do Prey Step:
    Try a prey step.
  If the terminal condition is met (e.g., maximum iterations or convergence criteria):
    Output the best feature selection result.
    Otherwise:
Repeat from step 2.
Return the best-selected feature set based on fish swarm optimization.

After selecting the optimal features, the features are fed into the DSTF-Net for advanced processing and accurate classification.

3.3 Deep SpatioTemporal Fusion Network (DSTF-Net)

Recently, a redesign of 2D CNNs, known as 1D CNNs, was developed. One of the popular DL methods is the 1-D CNN, which is made up of pooling and convolution layers. The architecture for Stacked CNN is depicted in Fig. 2. It is standard procedure to use the input through layers, neurons, and activation procedures using the ReLU. Usually, nonlinearity is introduced by such activation. To avoid overfitting, layers of dropouts and normalization methods are typically employed. A 1D CNN requires substantially fewer processing resources than a 2D CNN with the identical structure, network, and hyperparameters. `

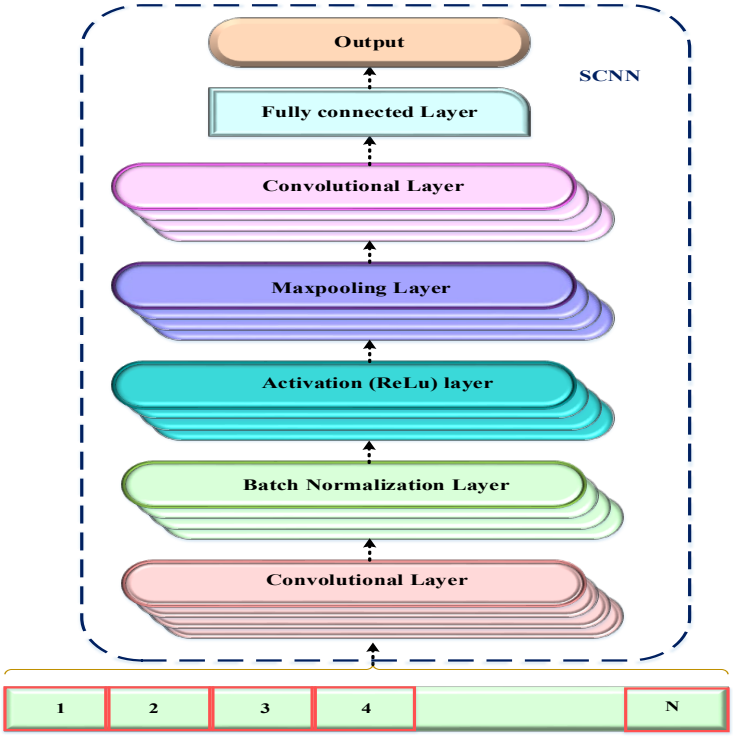

Fig. 2: SCNN architecture

According to more recent studies, most 1D CNN applications, which frequently employ micro topologies, usually employ networks (with one or two hidden CNN layers) of 50 neurons or fewer. Smaller 1D CNNs are especially appropriate for instantaneously, inexpensive tasks because of their lower processing needs, particularly on portable computing devices. The research specifically focused on applications with significant signal variations and inadequate categorization, such as time-series forecasting, civil, high-power circuits, power plants or motors, patient ECGs, etc., and found that compact 1D CNNs demonstrated superior performance. The main difference between 1D-CNN and 2D-CNN is that the latter uses 1D arrays as input vectors rather than the matrices that are often used in 2D-CNNs.The following are the fundamental mathematical equations (9) needed to operate a CNN:

$$a_{o,fl}^{l} = f\left(\sum_{im} a_{i}^{l-1} * k_{io,fl}^{l} + b^{l}\right) \tag{9}$$

The result of the layer l is represented by $a_{o,fl}^{l}$, where l is the layer index. The input a_{i}^{l-1} Convolution correlates to the kernel $k_{io,fl}^{l}$, and the bias term b^{l} is summed by the activation function $f(.)$.

$$a_{o}^{l} = f\left[max\left(\sum_{im} a_{i}^{l-1}\right) + b^{l}\right] \tag{10}$$

Eq. (10), which explains exactly how the max pooling procedure operates during the lth layer. Here, a_{o}^{l} is the result of applying max pooling with relation to the input overall from the preceding layer $(l-1)$, then applying the activation function $f(.)$ and adding the bias term b^{l}.

$$a_{o}^{l} = f\left(a_{i}^{l-1} * c_{io}^{l} + b^{l}\right) \tag{11}$$

The lth layer's convolution procedure is represented by Eq. (11). Using the kernel c_{io}^{l} to convolve the input from the $(l-1)$th layer, a_{i}^{l-1}, adding the bias b^{l}, and applying the activation function $f(.)$, the output a_{o}^{l} is produced.

It is possible to learn the parameters. b and c In this framework, as well as the number of filters (F) in every layer. Computational efficiency is achieved by using 1D-CNNs, which enable 1D arrays' linear weighted summation. The

accuracy and computing cost of the model are improved by these simultaneous actions that are performed during the forward and backpropagation phases.

When analyzing time series data, the RNN model's return loop enables it to effectively utilize previous knowledge. However, an RNN has information and storage limitations. The gradient vanishes due to its poor ability to recognize long-term dependence. To mitigate, the LSTM was created to address the RNN's weaknesses. The cells in the memory that hold a gate function that regulates long-term past information serve as the LSTM structure's foundation. Three distinct kinds of gates are present in an LSTM system: input-gate i_t, forget-gate f_n, and output-gate o_n. These three gates are shown in Fig. 3.

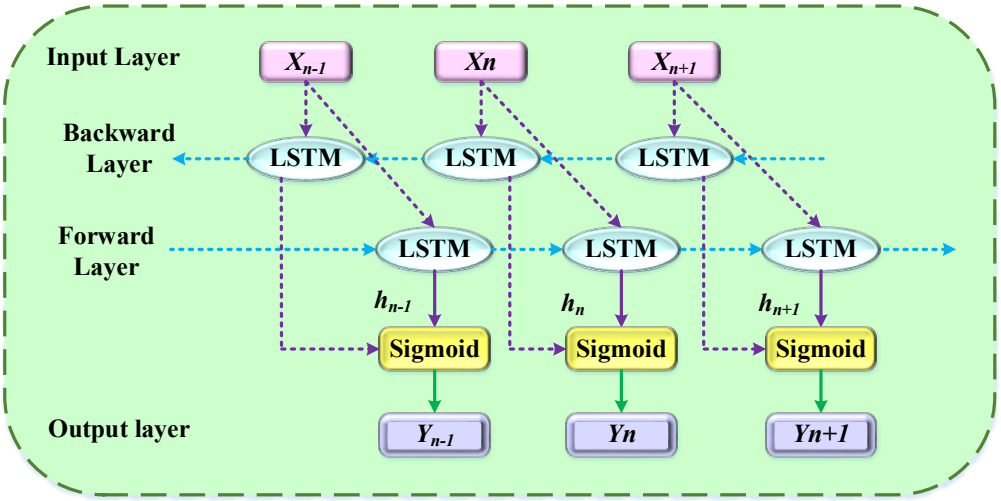

Fig. 3: BiLSTM architecture

Each gate in an LSTM network uses sigmoid functions and point-wise multiplication to control the memory cells' condition. The gates handle the resultant information. h_{n-1} from the preceding layer's hidden state and the current condition of the input data x_n. The "forget gate" determines if data should be retained or deleted. It uses a sigmoid function to analyse the current input. x_n as well as the previously concealed state h_{n-1}. The forget gate's output, represented by the symbol f_n, fluctuates between zero and one. A value near one signifies that the information will be reserved, whereas a value around zero suggests that the information will be deleted. The following equation (12) represents this mechanism:

$$f_n = \sigma(W^f.[h_{n-1}.x_n] + b_f) \tag{12}$$

Which additional data must be contributed to the cell state is then decided by the "input gate." Additionally, it employs the sigmoid function, which yields values between 0 and 1. The function of the input gate is explained as follows, as equation (13):

$$i_n = \sigma(W_i.[h_{n-1}.x_n] + b_f) \tag{13}$$

After processing the hidden state h_{n-1} and the current input x_n, the tanh function generates a vector of new candidate values, $\bar{C}_n$, that might be appended to the state, as given in equation (14):

$$\bar{C}_n = tanh(W_c.[h_{n-1}.x_n] + b_c) \tag{14}$$

The previous state C_{n-1} is multiplied by f_n to update the cell state C_n. The product of i_n and $\bar{C}_n$ is then added, and is given as equation (15):

$$C_n = (f \odot C_{n-1}) + (i_n \odot \bar{C}_n) \tag{15}$$

The hyperbolic tangent activation function is represented by $tanh$ in this equation, whereas ⊙ denotes element-wise multiplication. Ultimately, the next hidden state h_n is determined by the "output gate." It takes into account both the output from the preceding layer and the cell state are given as equations (16) and (17):

$$o_n = \sigma(W_0.[h_{n-1}.x_n] + b_0) \tag{16}$$

$$h_n = o_n \tanh \odot(C_n) \tag{17}$$

The output gate's bias and weights are represented by W_0 and b_0 in these final equations. The ultimate output is determined by the output gate h_n, the subsequent hidden state, using the sigmoid function σ and the $tanh$ function.

To put it briefly, a typical LSTM network is limited to using the data it has previously come across in the sequence. In contrast, the Bi-LSTM architecture has two LSTM layers: a forward LSTM that functions to handle the input information sequence and a backward LSTM that does the opposite. The Bi-LSTM's architectural diagram is seen in Fig. 3. While the backward LSTM layer gathers data from a later date to the current time step, the forward LSTM layer records data from earlier times to the current time step. Next, the outputs from the two hidden layers in each time step are combined. This arrangement guarantees that the hidden state h_n in the Bi-LSTM network has information from both the forward and backward directions at any given time n. The sum of the output elements from each of these orientations serves as a symbolic representation of this. Because of its ability to incorporate input from both past and future contexts, the Bi-LSTM architecture performs better than traditional LSTM and RNN models, particularly in situations where bidirectional context awareness is crucial.

The DSTF-Net is a useful technique for obtaining extremely high features of the series of input while working with large-scale, rapidly changing data. Because of the hierarchical representation it creates, the DSTF-Net is excellent at seeing intricate patterns in the data. By using enhanced features built on prior layers, each succeeding layer in SCNN improves the detection of complex intrusions. Bi-LSTM facilitates extracting spatial knowledge. Using two uncommon LSTM techniques: one that examines the information series from the starting point to the end, and the other that does so from the very end to the starting point, Bi-LSTM can preserve temporal interactions both forward and backward. By providing each data point with a thorough context through bidirectional processing, the model is able to identify the whole range of temporal patterns present in the data. Because intrusions such as port scans will continually evolve, having this type of ability is beneficial for monitoring for changes in those intrusions. A strong IDM is achieved through a partnership between DSTF-Net and others.

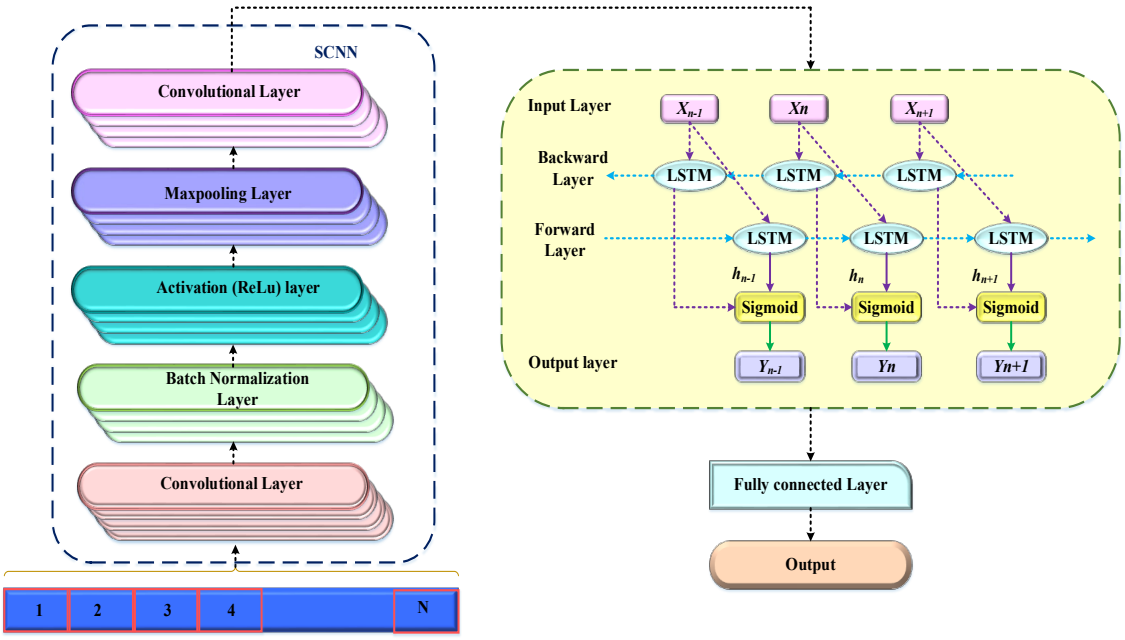

Fig. 4: Architecture of the DSTF-Net model

The DSTF-Net model is an excellent match for the IDS Implementation in IoT Networks in Fig. 4. This model provides detailed information regarding both geographic locations and times to determine whether an intrusion occurred or not. Due to the limited computing capabilities that many IoT networks operate under, the comprehensive nature of the model allows for a large number of features to be accounted for and also gives a more thorough analysis than most other models.

4.0 EXPERIMENTAL SETUP AND PERFORMANCE EVALUATION

This section will provide an assessment of the developed deep learning Model for IoT Intrusion Detection and its proposed application. The model performs exceptionally well in detecting malicious and unauthorized activities across all types of IoT Networks by identifying complex, time-variant, and spatio-temporal patterns in the data traffic generated by IoT devices. With high detection accuracy, the model allows for meaningful interpretation of results along with enhanced trust in the reliability of the results, which will be critically important for implementing an IDS within real-world IoT use cases like Smart Home, Industrial IoT Platforms (IIoT), and Critical Infrastructure Security (CIS) Frameworks.

4.1 System tools and configuration

The IoT attack detection framework, as described, has been developed with the aid of Python 3.10 using Jupyter Notebook as the main IDE. TensorFlow has been used for creating & training various models, while Scikit-learn provided data pre-processing, statistical analysis, and assessing performance metrics of each model. The experiments were conducted on a 64-bit Windows 10 machine equipped with a 7th-gen Intel Core i7 processor (2.8 GHz), which had sufficient computational power for both the training and testing phases of the study. The IOT attack dataset was divided into training (80%) and testing (20%) to enhance the potential for generalising to previously unseen attack types. This system will provide researchers with a reliable environment to evaluate the effectiveness of more sophisticated deep learning networks at identifying complex intrusions within an IoT network.

4.2 Dataset Description

4.2.1 Dataset 1: UNSW-NB15 dataset

In 2015, the original paper was published with over 2,500,000 simulated network packets that comprise the UNSW-NB15 dataset. It contains 9 attack classes and 1 non-anomalous class of packets, which are: exploits, reconnaissance, DoS, Generic, Shellcode, Fuzzers, Backdoors, Worms, and Analysis. The dataset is very unbalanced, with over 87% of the packets being non-anomalous. Table 2 provides a thorough breakdown of packets.

Table 2: Class distribution of UNSW-NB15 dataset

Class	No.of records	% of total data
Non-anomalous (normal)	2,218,716	87.35
Exploits	44,525	1.75
Reconnaissance	13,987	0.55
DoS	16,353	0.64
Generic	215,481	8.48
Shellcode	1,511	0.06
Fuzzers	24,246	0.95
Analysis	2,677	0.11
Backdoor	2,329	0.1
Worms	174	0.01
Total	2,540,044	

4.2.2 Dataset 2: TON-IOT dataset

The ToN-IoT dataset, which combines several data sources gathered throughout a full Industrial Internet of Things (IIoT) network, is the second dataset used in this study. It contains network connection information, system logs from Linux and Windows operating systems, and sensor data from linked nodes. The ToN-IoT repository makes the data, which were collected using a medium-scale IoT testbed set up at the University of New South Wales (UNSW) Canberra Cyber Range Labs, publicly accessible [24]. The ToN-IoT datasets are provided in CSV format, containing a dedicated column that identifies each record as either *attack* or *benign*, along with an *attack-type* subclass that specifies the nature of the attack. These cyberattacks were intentionally executed and captured within the IIoT environment, targeting various IoT and IIoT sensors. Scanning, injection, denial of service (DoS), distributed denial of service (DDoS), man-in-the-middle, backdoor, cross-site scripting, ransomware, and password cracking are the nine types of attacks identified in this dataset. The ToN-IoT network dataset consists of 44 attributes per instance of data and a label indicating "attack" or "normal" for each example, according to the attack type applied to it. The total distribution of attack versus normal instances included in both training and testing portions is provided in Table 3.

Table 3: Description of ToN-IoT Network dataset

Attack Type	No. of records
Back door	20,000
DDoS	20,000
DoS	20,000
Injection	20,000
Mitm	1,043
Password	20,000
Ransomware	20,000
Scanning	20,000
Xss	20,000
Normal	300,000
Total	**461,043**

4.2.3 Dataset 3: IoT-23 dataset

The IoT-23 project contains all of the network traffic collected from devices used for data collection. The IoT-23 is the database for the third dataset being used in this paper. It contains 23 different types of network capture incidents. Out of these, 20 include malware-infected traffic, and 3 represent benign (good, normal) activity for IoT devices. The dataset was created in January 2020 with all of the original network activity from 2018-2019, and was initially developed by the Stratosphere Research Group and AIC Group (Czech Technical University) based on information received from the Czech Republic government. Since the IoT-23 dataset provides real examples of both good and bad traffic from IoT devices, it provides a benchmark for researchers that is used for developing and evaluating machine intelligence intrusion detection and malware detection. The benign (good, normal) traffic was captured from real devices: a Philips HUE smart bulb, an Amazon Echo Smart Assistant, and a Somfy Smart Door Lock, it can be assured that the data contains real device information and is not based on simulated environments. Every packet capture (PCAP) file gets its own analysis in addition to its own labels assigned by the Flaber Python software. The resulting 23 labelled data sets are converted to individual Pandas DataFrames. In order to provide maximum computational efficiency, the first 10 rows from each PCAP file and then the next 100,000 rows from each PCAP file have been removed. These separate DataFrames are then combined into 1 dataset. The (processed) reduced size of 8.8 GB was created as a lightweight version of the original IoT-23 data set to enable efficient analysis and experiments.

4.3 Hyperparameter Configuration

To optimize the performance and generalizability of the recommended framework, the hyperparameters of the framework have been optimally tuned through both grid and random search techniques, as shown in Table 4. Each component's final tuning parameters/ configuration settings have been consolidated into Table 4.

Table 4: Hyperparameter Configuration of the proposed framework

Module	Parameter	Value / Configuration
ANDPM	Kernel Type	RBF
	Regularization Parameter (C)	1.0
	Gamma	0.1
	Imputation Iterations	50
	Synthetic Sampling Ratio	1:1 (Attack: Benign)
	Cross-Validation	5-fold
	Stopping Criterion	≤ 1e-4
ADFO	Population Size	30
	Maximum Iterations	100
	Visual Distance (V)	0.5
	Step Size (S)	0.3
	Crowd Factor (δ)	0.6
	Prey Probability	0.8
	Assimilation Rate	0.4
	Fitness Function	Mutual Information
	Stopping Condition	No improvement in 10 iterations
DSTF-Net	Input Dimension	Optimized feature subset (post-ADFO)
	Convolution Layers	2 (64, 128 filters)
	Kernel Size	3 × 3
	Activation Function	ReLU
	Pooling Layer	MaxPooling (2 × 2)
	Dropout Rate	0.3
	Bi-LSTM Layers	2 (128 units each)
	Recurrent Dropout	0.2
	Optimizer	Adam
	Learning Rate	0.001
	Batch Size	64
	Epochs	50
	Loss Function	Categorical Cross-Entropy
	Output Layer	Softmax

4.4 Fitness Convergence Analysis of ADFO

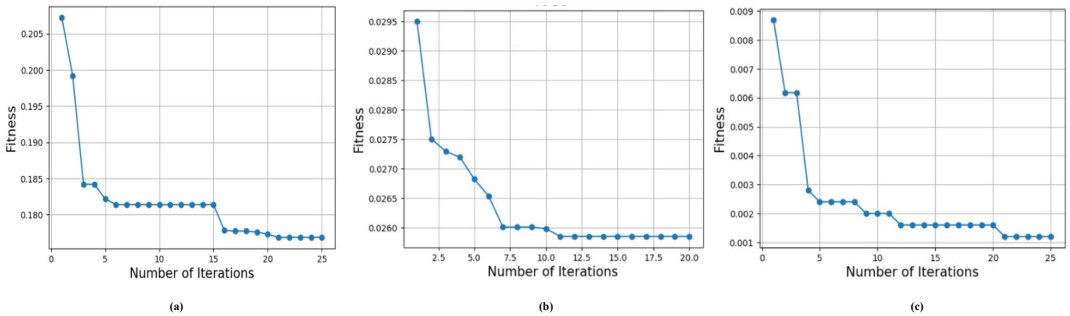

Fig. 5: Fitness convergence curve of the ADFO algorithm showing the variation of fitness value with respect to the number of iterations using (a) UNSW-NB15 dataset, (b) ToN-IoT dataset, (c) IoT-23 dataset

The fitness convergence behavior of the ADFO algorithm on the three datasets: (a) UNSW-NB15, (b) ToN-IoT, (c) IoT-23, is shown in Fig. 5. In every case, the fitness value, quite steeply, goes down during the initial iterations, which indicates the swift exploration along the way to effective identification of the relevant features. After around the 10th iteration, the fitness curves become flat, which signifies convergence towards the optimal feature subset with very little fluctuation. This shows the power and the stability of the ADFO algorithm in getting quality feature selection across the varied datasets of IoT networks.

4.5 Performance Evaluation

The proposed Adaptive Spatiotemporal Defense framework's performance has been evaluated through the use of standard classification metrics that come from the confusion matrix. The formulas are as follows:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{18}$$

$$Precision = \frac{TP}{TP+FP} \tag{19}$$

$$Recall = \frac{TP}{TP+FN} \tag{20}$$

$$Specificity = \frac{TN}{TN+FP} \tag{21}$$

$$F1-score = 2 \times \frac{Precision \times Recall}{Precision+Recall} \tag{22}$$

$$Macro\ F1-score = \frac{1}{N}\sum_{i=1}^{N} F1_i \tag{23}$$

where N is the number of classes, and $F1_i$ denotes the F1-score calculated for each class separately.

5.0 RESULTS

5.1 Cross-validation Evaluation

The proposed intrusion detection model's reliability and generalization ability were validated with the 5-fold cross-validation method on the UNSW-NB15, ToN-IoT, and IoT-23 datasets. This technique divided up each dataset into k equal parts, and the model was trained using (k–1) of these parts and tested with the remaining one. In order to minimize the bias and variation, this procedure was repeated k times, and the average performance was determined. This validation approach has shown that the model could maintain its stability amid the different patterns of IoT network traffic and exhibited consistent performance and resistance to overfitting in all three datasets. The cross-validation results of the proposed model evaluated on the three datasets are shown in Tables 5(a), 5(b), and 5(c).

Table 5(a): 5-fold cross validation using UNSW-NB15 Dataset

Fold	Accuracy	Precision	Recall	Macro F1	Weighted F1	Specificity
Fold 1	0.8310	0.7324	0.5051	0.5154	0.8032	0.9794
Fold 2	0.8408	0.7323	0.5498	0.5753	0.8240	0.9808
Fold 3	0.8338	0.7068	0.5409	0.5529	0.8147	0.9801
Fold 4	0.8396	0.7077	0.5448	0.5581	0.8195	0.9808
Fold 5	0.8346	0.6743	0.5461	0.5585	0.8189	0.9802
Average	0.8360	0.7107	0.5373	0.5520	0.8161	0.9803

Table 5(b): 5-fold cross-validation using ToN-IoT Dataset

Fold	Accuracy	Precision	Recall	Macro F1	Weighted F1	Specificity
Fold 1	0.9706	0.8831	0.9379	0.9037	0.9712	0.9966
Fold 2	0.9740	0.9206	0.9231	0.9184	0.9744	0.9969
Fold 3	0.9630	0.9049	0.8953	0.8971	0.9632	0.9949
Fold 4	0.9740	0.9045	0.9280	0.9134	0.9745	0.9969
Fold 5	0.9694	0.8888	0.9232	0.9025	0.9700	0.9961
Average	0.9702	0.9004	0.9215	0.9070	0.9707	0.9963

Table 5(c): 5-fold cross validation using IoT-23 Dataset

Fold	Accuracy	Precision	Recall	Macro F1	Weighted F1	Specificity
Fold 1	0.8943	0.7404	0.7117	0.6965	0.8771	0.9689
Fold 2	0.6887	0.7061	0.6282	0.5893	0.6146	0.9038
Fold 3	0.7897	0.6918	0.6930	0.6571	0.7800	0.9524
Fold 4	0.8958	0.7255	0.7381	0.6953	0.8775	0.9692
Fold 5	0.6944	0.6633	0.6599	0.6164	0.6935	0.9361
Average	0.7926	0.7054	0.6862	0.6509	0.7685	0.9461

As shown in Figs. 6,7, and 8, the training and testing curves illustrate the learning dynamics and convergence stability of the proposed model over 50 epochs. The remarkable increase in the accuracies of both training and testing in the initial epochs, followed by a stabilization around the 20-epoch mark, can be taken as an indication of successful learning. The loss function curve tells the same story; it drops significantly at first and then keeps approaching the minimal value very slowly, thus indicating effective optimization and stable behavior of the model. The training and testing curves being very close to each other reveal the model's strong generalization ability with minor overfitting. Moreover, the proposed solution's resilience, scalability, and flexibility in dealing with different IoT intrusion detection scenarios were proven by the consistently high accuracy and low loss values across all three benchmark datasets.

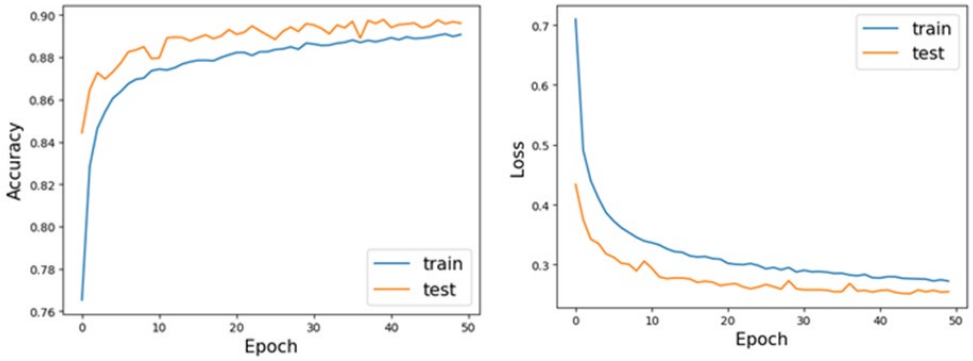

Fig. 6: Accuracy and loss graphs of the proposed method using the UNSW-NB15 dataset

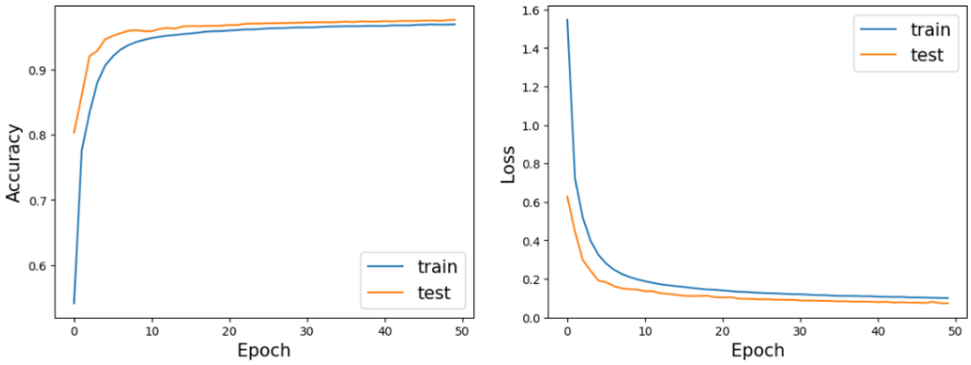

Fig. 7: Accuracy and loss graphs of the proposed method using the ToN-IoT dataset

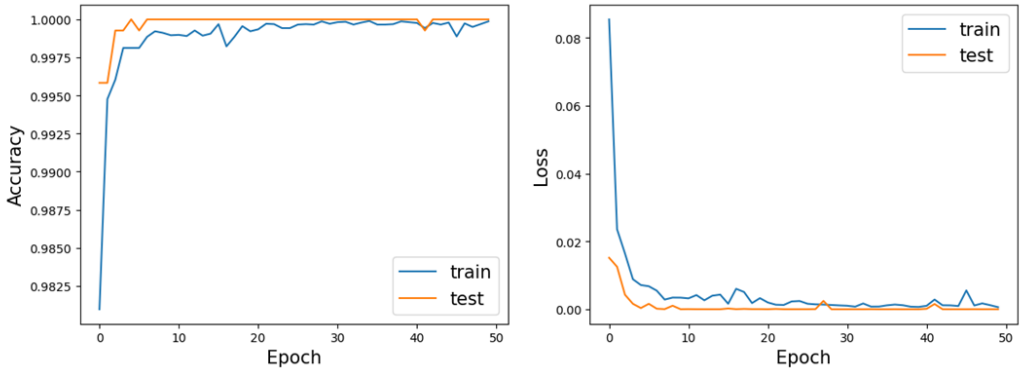

Fig. 8: Accuracy and loss graphs of the proposed method using the IoT23 dataset

5.2 Confusion Matrix

By the confusion matrices, the proposed model's classification performance over different classes is evidenced. In all situations majority of the samples can be found on the diagonal that signifies correct predictions with little misclassification. The considerable diagonal dominance is the proof of the model's capability to differentiate between different assault types and normal traffic. The thereby demonstrated model's ability to cut down false positives and negatives, thus, assuring reliable detection performance, is supported by the relatively small off-diagonal values. In short, the results give confirmation to the strength and the effectiveness of the proposed method in accurately identifying different network traffic types in intrusion detection tasks. The confusion matrix of the proposed model is given in Fig. 9 (a) UNSW-NB15, (b) ToN-IoT (c) IoT-23 datasets.

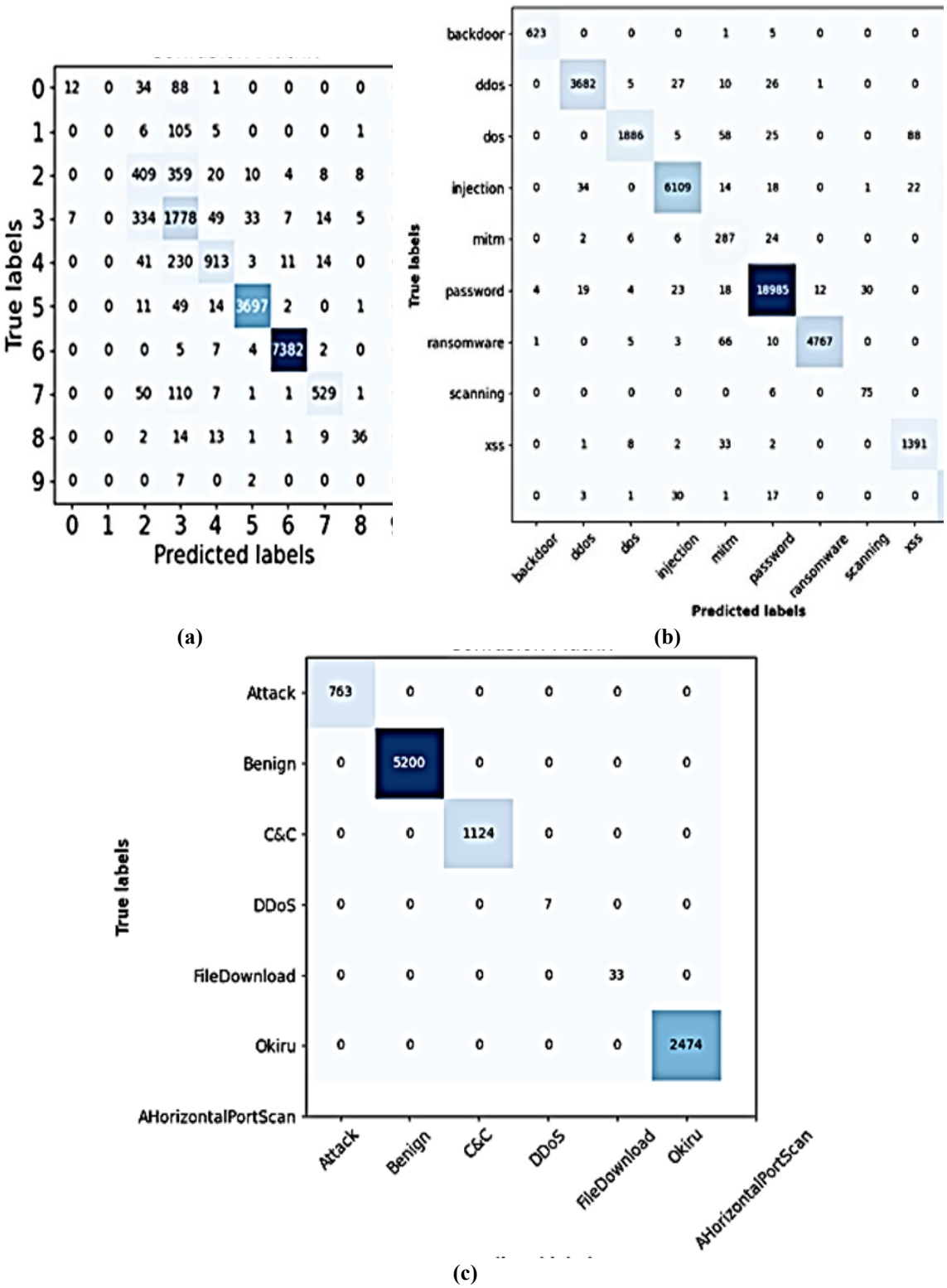

Fig. 9: Confusion matrix of the proposed approach (a) UNSW-NB15 (b)ToN-IoT (c) IoT-23 datasets

5.3 ROC Curve

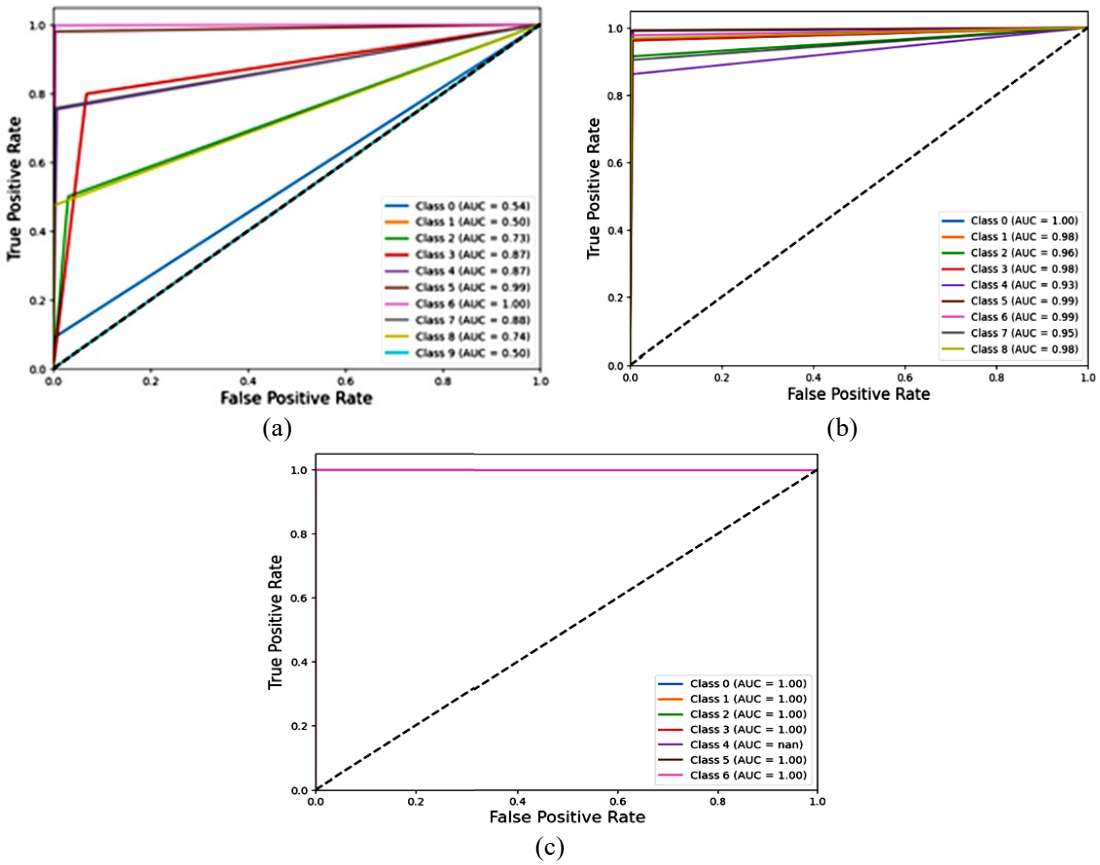

Fig. 10: ROC curve of the proposed model using (a) UNSW-NB15, (b) ToN-IoT, (c) IoT-23 datasets

The three datasets show varying performance for the proposed method, as demonstrated by the ROC curves shown in Fig. 10(a)-(c). In the UNSW-NB15 dataset (Fig. 10(a)), each class has a high AUC (0.76-0.94) and demonstrates good class-level discrimination. Class 8 has the highest level of separation (AUC = 0.94), and Class 4 (AUC = 0.92) is next in performance. Class 0 (AUC = 0.76) has relatively low separation and indicates that there is a larger degree of difficulty in classifying attacks than in other categories. In Fig. 10(b) (ToN-IoT), almost all classes show a high AUC value that is consistently between 0.98 and 1.00. There is a very good generalization ability of the classifier (excellent ability to distinguish between benign and malicious traffic) in a heterogeneous IoT environment. The AUC values of the IoT-23 dataset (Fig. 10c) shows all the values are equal to one (1.0 AUC), meaning perfect separation between the target sample (in this case the IoT device subtypes), however, it is important to understand the dataset characteristics are contributing to these results, and since this dataset has very easy to separate classes (subtypes of IoT devices) and has a similar or very simple feature distributions or patterns for each class, therefore it will require additional testing on actual datasets or environments that have complex or highly variable traffic types or classes before concluding that the model has good generalization capabilities. Table 6 provides the overall performance for the three datasets being compared.

Table 6: Overall performance metrics

Metrics (%)	UNSW-NB15 dataset	ToN-IoT dataset	IoT-23 dataset
Accuracy	89.61	97.94	99.99
Precision	89.69	97.85	99.99
Recall	89.61	97.62	99.99
F1-score	89.29	97.68	99.99
Sensitivity	89.61	99.99	99.99
Macro-F1	88.50	92.42	99.99

5.4 Inference Time and Computational Complexity Analysis

In order to prove the practical applicability of the proposed Adaptive Spatiotemporal Defense Framework, the performance during inference as well as the complexity during inference are analyzed for deployment in IoT environments. Although the model exhibits high accuracy in detecting attacks on benchmark datasets (89.61% on UNSW-NB15, 97.94% on ToN-IoT, and 99.99% on IoT-23), practical deployment requires high performance in terms of inference speed and efficiency. The framework comprises ANDPM (data preprocessing), ADFO (feature optimization), and DSTF-Net. The ANDPM and ADFO are executed during offline training. The inference process mainly involves the DSTF-Net module. The dimensionality reduction in the input data by ADFO reduces the inference time. As a matter of fact, the empirical evaluation on a standard CPU-based environment (Intel i7, no GPU acceleration) indicates that the proposed model has low latency in the range of 3.8 ms, 4.5 ms, and 5.1 ms per sample for UNSW-NB15, ToN-IoT, and IoT-23 datasets, respectively. All these results are summarized in Table 7. The proposed model offers a small model size, approximately 1.2 million parameters, along with a low computational cost, estimated at approximately 0.5 GFLOPs, ensuring efficient execution even without the support of a specialized environment. The proposed framework, therefore, ensures the support of a near-real-time intrusion detection system without any considerable latency.

Table 7: Inference Time and Computational Efficiency Analysis

Dataset	Accuracy (%)	Avg. Inference Time (ms/sample)	Estimated Parameters (Millions)	FLOPs (GFLOPs)
UNSW-NB15	89.61	3.8 ms	1.2 M	~0.45
ToN-IoT	97.94	4.5 ms	1.2 M	~0.48
IoT-23	99.99	5.1 ms	1.2 M	~0.52

Based on the complexity analysis, the major factors that affect the complexity of the inference are the 1D-CNN and Bi-LSTM layers in DSTF-Net. The complexity of the convolutional layer is given as $O(n.k.f)$, where n is the number of input features, k is the kernel size, and f is the number of filters. The complexity of the Bi-LSTM layer is given as $O(n.h^2)$, where h is the hidden unit. Considering the optimized features, the input size is small, hence improving the efficiency of the inference. The space complexity is $O(p)$, where p is the number of parameters, and it is low. So, it is evident that with lower inference latency, reduced feature dimensionality, and time and space complexity, the proposed framework is apt for deployment in a real-time IoT edge environment, where a fair balance between detection and efficiency is required.

5.5 Comparative Analysis

The model proposed in this study has been evaluated against contemporary machine learning and deep learning techniques using the UNSW-NB15, ToN-IoT, and IoT-23 data sets. Using the UNSW-NB15 data set, comparisons were made to CNN, Spiking Neural Networks, Linear Neural Networks, and the LNN+NID Hybrid Framework. For the ToN-IoT data set, the model was evaluated against traditional classifiers such as Decision Tree, Multinomial Logistic Regression, Resilient Backpropagation, Logistic Regression, Voting, and Stacking approaches. Comparisons made with the IoT-23 dataset include Decision Tree, CNN-1D, Random Forest, and SVM techniques. The evaluation

process used accuracy, precision, recall, and F1-score to measure Detection Capability and Generalization Performance. Overall, the findings show that the proposed framework consistently outperforms the other models with respect to performance and durability.

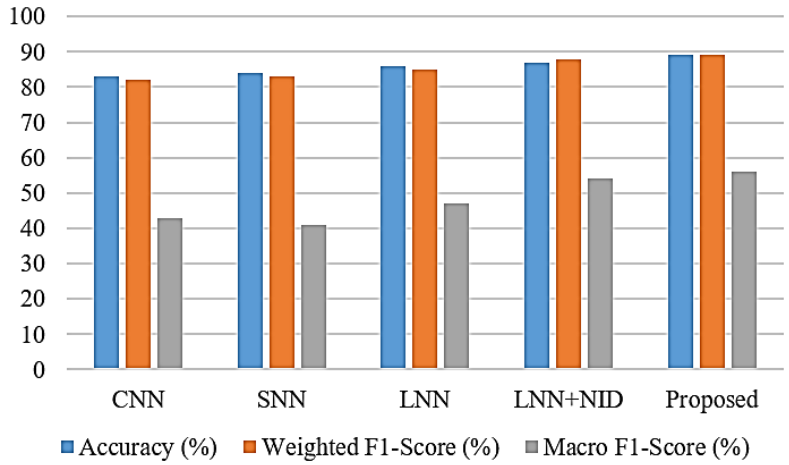

Fig. 11: Comparison of existing and proposed models on UNSW-NB15

According to the data presented in Fig. 11, our proposed model is superior to both the CNN and the SVM, as well as to the LNN-NID model, in terms of performance, on the UNSW-NB15 dataset when evaluating the three primary evaluation metrics (accuracy, weighted F1 score, and macro F1-score). The CNN and SVM were clearly at the bottom of the performance spectrum, as indicated by their poor performance on all three of these metrics. However, our proposed model consistently produced the highest scores of all three evaluation criteria; thus, supporting our claim that our proposed model is the best performing model in terms of providing highly accurate and well-balanced classifications of both major and minor classes of data, because of its outstanding unweighted and macro F1 Score. The ability of our proposed model to effectively deal with the degree of complexity and class imbalance associated with the UNSW-NB15 intrusion dataset has been proven. A comparison of our proposed model against all other existing models was made using the UNSW-NB15 dataset, as shown in Table 8.

Table 8: Comparison of the proposed method with existing models

Model	Accuracy (%)	Weighted F1-Score (%)	Macro F1-Score (%)
CNN	83	82	43
SNN	84	83	41
LNN	86	85	47
LNN+NID	87	88	54
Proposed	89	89	56

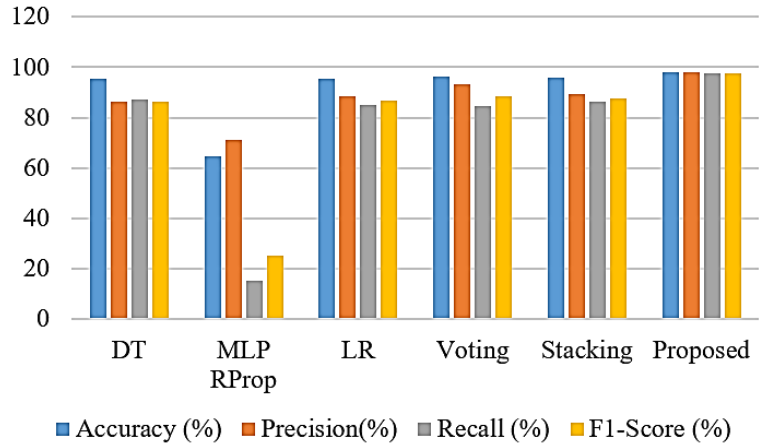

Fig. 12: Comparison of existing and proposed models on ToN-IoT

All four key metrics for the ToN-IoT dataset show how much better the proposed model is compared to all baseline models (DT, MLP, RProp, LR, Voting, Stacking) in Fig 12. For the majority of the baseline models, performance has shown a significant drop when compared to the proposed model. The proposed model outperforms all baseline models on all four key metrics, indicating that it has the best classification capability, handles class imbalance well, and generalizes the best for intrusion detection. The comparison values of the proposed model versus the current model for the ToN-IoT dataset can be found in Table 9.

Table 9: Comparison using ToN-IoT dataset

Methods	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
DT	95.551	86.263	87.344	86.290
MLP RProp	64.714	71.005	15.314	25.194
LR	95.312	88.410	84.980	86.661
Voting	96.321	93.119	84.555	88.631
Stacking	95.757	89.199	86.515	87.836
Proposed	97.94	97.85	97.62	97.68

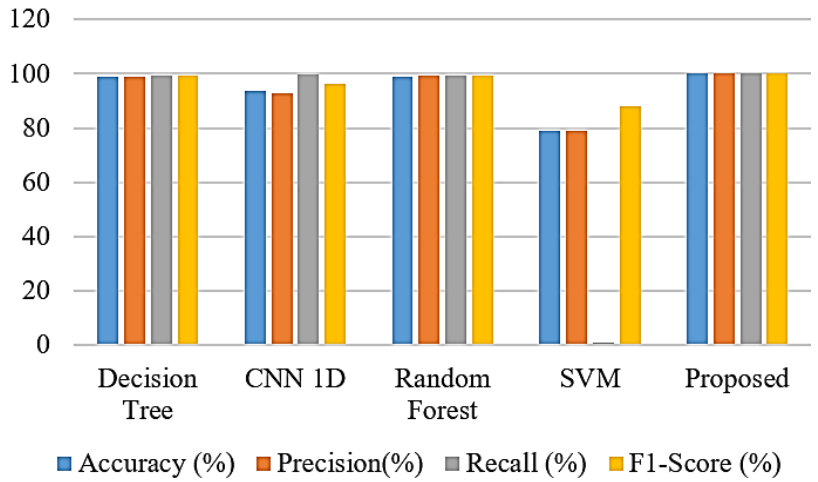

Fig. 13: Comparison of the existing model with the proposed approach using the IoT-23 dataset

Fig. 13 shows that the proposed model consistently outperforms traditional baseline classifiers, including Decision Tree, CNN-1D, Random Forest, and SVM, across all four Evaluation Criteria. Many existing classifiers show a significant drop in performance, while others, especially SVM, demonstrate a drastic decrease in both Recall and F1-Score. The proposed framework performs well across all metrics, with strong values approaching unity. Consequently, this indicates that the proposed framework can detect attacks accurately, generate very few false positives, and generalize well to different types of IoT attacks. Thus, the results provide evidence of how well the proposed model adapts to the complex multi-class features present within the IoT-23 Dataset. The summary of the performance comparison between the proposed framework and other existing models can be found in Table 10.

Table 10: Comparison of the existing model with the proposed model using the IoT-23 dataset

Methods	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Decision Tree	98.79	98.94	99.53	99.24
CNN 1D	93.90	93.0	99.76	96.27
Random Forest	98.91	99.36	99.26	99.31
SVM	78.87	78.88	1.00	88.19
Proposed	99.99	99.99	99.99	99.99

5.6 Ablation Study

An exhaustive ablation analysis was conducted to analyze the differences produced by each of the various parts of the proposed Adaptive SpatioTemporal Defense Framework; thus, the ablation assessment studied the effect on the performance of the model when key parts of the framework, ANDPM (Adaptive Noised Dynamic Position and Motion), ADFO (Adaptive Detection Feature Optimization), and DSTF-Net (Dynamic SpatioTemporal Learning Network) were taken away or substituted. Because comparisons of these altered frameworks with the whole framework were completed, the results of the ablation analysis illustrate the contributions of each of the key modules to the precision of detection, the enhancements to the extraction of features, the balance of the data, and the capability for spatio-temporal learning. It was concluded that all three modules contribute significantly to increasing classification accuracy, enhancing the distribution quality (i.e., how well the samples represent the range of possibilities) of the data, and enhancing the adaptability of the framework to the many different forms of intrusion that could be experienced by an IoT system. The results of the ablation experiment performed with the three datasets used to evaluate the proposed Adaptive Spatiotemporal Defense Framework are presented in Table 11.

Table 11: Ablation Experiment

Variant (removed/replaced)	UNSW-NB15	ToN-IoT	IoT-23
Full model (ANDPM + ADFO + DSTF-Net)	89.61 / 89.29	97.94 / 97.68	99.99 / 99.99
Remove ANDPM (use mean imputation + no adaptive sampling)	84.20 / 83.80	95.10 / 94.80	99.50 / 99.50
Remove ADFO (use all features, no optimization)	86.75 / 86.40	96.30 / 96.05	99.70 / 99.70
Replace DSTF-Net with ConvNet only	82.40 / 81.90	93.80 / 93.40	99.20 / 99.20
Replace ANDPM with SMOTE + mean impute	87.10 / 86.80	96.80 / 96.50	99.80 / 99.80
Replace ADFO with PCA (95% var.)	88.00 / 87.70	97.20 / 97.00	99.95 / 99.95

6.0 DISCUSSION

The experiment's results demonstrate that the proposed Adaptive Spatiotemporal Defense Framework consistently has good performance across all assessment measures, confirming that the framework is effective in dealing with the complexities and diversity of IoT traffic. The performance of the proposed model is significantly better than that of existing methods, including traditional classifiers and prior deep learning models, especially in terms of Recall and F1-score, where most of these approaches have difficulties due to issues related to class imbalance and their inability to model temporal data accurately. Additionally, through statistical validation using paired t-tests, the performance improvements for all of the proposed models across the three datasets are confirmed to be statistically significant ($p < 0.05$). Although the IoT-23 dataset reports a near-perfect accuracy of 99.99%, it is important to be cautious in interpreting these results. The high accuracy may be an artifact of the dataset's characteristics. Thus, it is important to validate the results to confirm the ability of the model to generalize and be robust to unknown attack patterns. The proposed framework is much more robust and reliable than the baseline methods. Additionally, upon performing a class-wise analysis, it was found that a few classes of minority attack types had lower recall and F1-scores than others, with the largest differences being found in the UNSW-NB15 dataset (e.g., classes 1, 8, and 9). This is primarily due to the very low number of instances in these classes, which leads to limited patterns available to train on. Also, there are often overlaps between the behaviors of rare attacks with benign traffic, resulting in a greater likelihood of misclassification. Therefore, this limitation of the framework is primarily due to the inherent challenges of working with materially imbalanced IoT datasets and is not a reflection of the proposed framework's structure. The overall advantages of our framework are derived from balanced nonlinear preprocessing, discriminative feature optimization, and a design for deep spatio-temporal learning, which create strong recognition capabilities, generalization abilities to previously unseen attacks, and efficient functionality under constraints imposed by IoT-edge devices. An interesting point of reference is the significant drop in Recall in SVMs, despite having high Precision ratings; this suggests that the classical model's tendency to overfit to dominant classes prevents adequate representation of the various types of minority attacks. While our method addresses this limitation, future research will need to evaluate how our method performs with highly obfuscated and/or encrypted traffic. Moreover, our results have been statistically validated and demonstrate that the proposed framework can be effectively applied in a variety of real-time ID applications for IoT devices with limited resources.

7. CONCLUSION

The Adaptive Spatiotemporal Defense Framework provides evidence that reliable ID within IoT environments can be accomplished without sacrificing efficiency by using the complementary learning stages of the framework. The framework combines these learning stages to improve the quality of information and to create a more accurate understanding of features across time for diverse intrusion types while still being deployable even with resource constraints. The framework's ability to integrate preprocessing, adaptive feature refinement, and spatiotemporal modeling is its primary innovation, and it is practical and scalable as a defense mechanism for heterogeneous IoT environments. Its promise for real-world application is supported by the consistent and good performance across multiple assessment metrics. Future enhancements could provide additional privacy-preserving and distributed learning capabilities to enable collaborative detection within large-scale IoT networks. Additional expansion possibilities include online model adaptation for emerging threats, providing support for encrypted traffic, and investigating lighter-weight architecture alternatives (e.g., transformer-based architectures) optimized for edge-device hardware. Such paths can increase the adaptability and operational readiness of the framework to serve next-generation IoT security solutions. The Adaptive Spatiotemporal Defense Framework establishes and maintains a high and stable performance level when it comes to the IoT intrusion detection system; however, there are limitations to the framework that should be investigated. First, although the framework provides a balanced approach to preprocessing, feature refinement, and spatiotemporal modelling, if the intrusion detection system encounters highly divergent attack types or attacks that have not been previously seen, the performance of the Adaptive Spatiotemporal Defense Framework may be negatively affected to some degree. This is especially true of dynamic IoT environments that are continuing to evolve, creating new kinds of threats and behaviors. Second, the framework is mainly designed to work with unencrypted traffic, which means its usefulness in networks that enforce data privacy through encryption is limited at present. Third, the Neural Network-based models that create the deep spatiotemporal models of Adaptive Spatiotemporal Defense Frameworks may present a challenge to the computational and memory-constrained edge devices, even when they have been optimized for use on resource-limited devices. Fourth, even though the preprocessing and feature selection stages enhance the detection rate of minority-class IoT devices, attacks

representing rare or "zero-day" attacks are a potential weakness due to insufficient representative data being available for training.

REFERENCES

[1] A. Heidari and M. A. Jabraeil Jamali, "Internet of Things intrusion detection systems: A comprehensive review and future directions," *Cluster Computing*, vol. 26, no. 6, pp. 3753–3780, 2023.

[2] S. F. Tan, A. Samsudin, and S. Alias, "Internet of Things: Security challenges and its future direction," in *Lecture Notes in Electrical Engineering*, vol. 547, pp. 483–488, 2019.

[3] I. Idrissi, M. Azizi, and O. Moussaoui, "IoT security with deep learning-based intrusion detection systems: A systematic literature review," in *Proc. 4th Int. Conf. Intelligent Computing in Data Sciences (ICDS)*, pp. 1–10, Oct. 2020.

[4] A. R. Khan *et al.*, "Deep learning for intrusion detection and security of Internet of Things (IoT): Current analysis, challenges, and possible solutions," *Security and Communication Networks*, vol. 2022, Art. no. 4016073, 2022.

[5] M. Azrour, J. Mabrouki, A. Guezzaz, and A. Kanwal, "Internet of things security: Challenges and key issues," *Security and Communication Networks*, vol. 2021, Art. no. 5533843, 2021.

[6] R. Ahmad and I. Alsmadi, "Machine learning approaches to IoT security: A systematic literature review," *Internet of Things*, vol. 14, p. 100365, 2021.

[7] A. D. Jurcut, P. Ranaweera, and L. Xu, "Introduction to IoT security," in *IoT Security: Advances in Authentication*, pp. 27–64, 2020.

[8] M. Anwer, S. M. Khan, and M. U. Farooq, "Attack detection in IoT using machine learning," *Engineering, Technology & Applied Science Research*, vol. 11, no. 3, pp. 7273–7278, 2021.

[9] N. Mishra and S. Pandya, "Internet of things applications, security challenges, attacks, intrusion detection, and future visions: A systematic review," *IEEE Access*, vol. 9, pp. 59353–59377, 2021.

[10] S. Alkadi, S. Al-Ahmadi, and M. M. B. Ismail, "Better safe than never: A survey on adversarial machine learning applications towards IoT environment," *Applied Sciences*, vol. 13, no. 10, p. 6001, 2023.

[11] J. Liu *et al.*, "Adversarial machine learning: A multilayer review of the state-of-the-art and challenges for wireless and mobile systems," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 123–159, 2021.

[12] I. H. Sarker, "Deep cybersecurity: A comprehensive overview from neural network and deep learning perspective," *SN Computer Science*, vol. 2, no. 3, p. 154, 2021.

[13] J. Asharf *et al.*, "A review of intrusion detection systems using machine and deep learning in internet of things: Challenges, solutions and future directions," *Electronics*, vol. 9, no. 7, p. 1177, 2020.

[14] H. Khazane *et al.*, "A holistic review of machine learning adversarial attacks in IoT networks," *Future Internet*, vol. 16, no. 1, p. 32, 2024.

[15] S. I. Popoola *et al.*, "Hybrid deep learning for botnet attack detection in the internet-of-things networks," *IEEE Internet of Things Journal*, vol. 8, no. 6, pp. 4944–4956, 2020.

[16] M. Zeeshan *et al.*, "Protocol-based deep intrusion detection for DoS and DDoS attacks using UNSW-NB15 and Bot-IoT datasets," *IEEE Access*, vol. 10, pp. 2269–2283, 2021.

[17] R. Zhao *et al.*, "A novel intrusion detection method based on lightweight neural network for internet of things," *IEEE Internet of Things Journal*, vol. 9, no. 12, pp. 9960–9972, 2021.

[18] B. Sharma, L. Sharma, C. Lal, and S. Roy, "Anomaly-based network intrusion detection for IoT attacks using deep learning technique," *Computers and Electrical Engineering*, vol. 107, p. 108626, 2023.

[19] S. Moualla, K. Khorzom, and A. Jafar, "Improving the performance of machine learning-based network intrusion detection systems on the UNSW-NB15 dataset," *Computational Intelligence and Neuroscience*, vol. 2021, Art. no. 5557577, 2021.

[20] M. Alsharaiah *et al.*, "An innovative network intrusion detection system (NIDS): Hierarchical deep learning model based on UNSW-NB15 dataset," *International Journal of Data and Network Science*, vol. 8, no. 2, pp. 709–722, 2024.

[21] M. A. Akif *et al.*, "Hybrid machine learning models for intrusion detection in IoT: Leveraging a real-world IoT dataset," *arXiv preprint arXiv:2502.12382*, 2025.

[22] P. R. Chithra Rani and K. Baalaji, "Deep learning-based ensemble stacking for enhanced intrusion detection in IoT-edge platforms," *Discover Applied Sciences*, vol. 7, no. 8, p. 928, 2025.

[23] N. Moustafa and J. Slay, "UNSW-NB15: A comprehensive data set for network intrusion detection systems," in *Proc. IEEE Mil. Commun. Inf. Syst. Conf. (MilCIS)*, pp. 1–6, Nov. 2015.

[24] N. Moustafa, “A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets,” *Sustainable Cities and Society*, vol. 72, p. 102994, Sep. 2021.
[25] S. Garcia, A. Parmisano, and M. J. Erquiaga, “IoT-23: A labeled dataset with malicious and benign IoT network traffic,” Zenodo, 2020.
[26] S. Hajla *et al.*, “Enhancing IoT network defense: Advanced intrusion detection via ensemble learning techniques,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 35, no. 3, pp. 2010–2020, 2024.
[27] B. Jayaraman *et al.*, “Detecting malicious IoT traffic using machine learning techniques,” *Romanian Journal of Information Technology and Automatic Control*, vol. 33, no. 4, pp. 47–58, 2023.