Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

TFF_ATNET: A TRIPLET FOLDED FEATURES LEARNING FRAMEWORK WITH AN ATTENTION-BASED TRANSFORMER NETWORK-BASED INTRUSION DETECTION SYSTEM

[1*]INDIRA, [2]DR. A. K. SAMPATH

[1*] Research Scholar, School of Computer Science and Engineering, Presidency University, Bangalore, India

[2] Professor, School of Computer Science and Engineering, Presidency University, Bangalore, India

Email Id: indira@pdit.ac.in* , sampath.ak@presidencyuniversity.in

ABSTRACT

Intrusion Detection Systems (IDS) play a vital role in protecting networks from malicious threats, yet they often face challenges such as imbalanced categories of traffic and distributional discrepancies between training and testing data. To overcome these limitations, this study introduces an integrated methodology that enhances IDS performance and robustness. The framework begins with data preprocessing and systematic data splitting through Apache Spark to handle imbalance and ensure efficient processing. For feature learning, a Triplet Feature Fusion module is employed, comprising Shallow Early-Stacked Feature Pooling (SESF), Deep Learning-based Statistical Features (DLSF), and Dual Statistical Features (DSF) to extract diverse and discriminative representations. These fused features are then processed by the Triplet Feature Fusion Attention Transformer Network (TFF_ATNet), where the Trans_Attention classifier captures long-range dependencies for precise classification. To further fine-tune the classification outcomes, the Self-Adaptive Walrus Optimization Algorithm (SA-WaOA) is applied, ensuring optimized detection performance. This model is implemented in Python and tested using three major datasets, namely CIC-IDS2017, UNSW-NB15, and TON_IoT. The evaluation of performance achieves an accuracy of 0.9993 on CIC-IDS2017 and low computation time (98 ms), superior sensitivity (0.9909), and specificity (0.9769), compared to existing methods. The proposed approach ensures balanced detection, improved classification efficiency, and contributes significantly to the advancement of IDS research in cybersecurity.

Keywords: *Intrusion Detection Systems, Feature Imbalance, Triplet Folded Feature Learning, Attention-based Transformer Network, Cybersecurity Measures.*

1 INTRODUCTION

Intrusion detection systems (IDS) are a critical aspect of network protection, offering real-time traffic surveillance to identify unauthorized activity and alert administrators of possible breaches [1]. Such functionality supports confidentiality, information integrity and availability in modern digital infrastructure [2]. Because of the rapid growth when it comes to the Internet of Things (IoT) and overall complexities of contemporary network environments, IDS has become a must-have tool utilized to protect against digital malicious activity changes [3]. Furthermore, Deep learning and machine learning (ML) techniques are also incorporated successfully in IDS, demonstrating their ability to analyze higher-dimensional traffic data and identify more complex intrusion patterns than classical rule-based systems [4].

Despite developments, there are numerous challenges to the accuracy and effectiveness of the contemporary IDS [5]. A popular challenge in the actual deployment of IDS is type imbalance of traffic, where uncommon types of attacks are scanty and, therefore, detection and false negatives are affected [6]. Misclassification also occurs when training and test data are of different distributions [7]. This distributional change leads to impacts on accuracy and model robustness [8]. Some of the crucial risk factors are explainability (decisions made by complicated models cannot be understood), excessive false positives that overwhelm the

administrators with too many alerts, and scalability and latency constraints that prevent detection in real time within the same network, which is extensive in size or dynamic [9]. All these challenge areas must be addressed for the accomplishment of an IDS that can fulfil the current cybersecurity demands [10]

To get rid of these challenges, several ML and deep learning methods have been investigated by experts [11]. CNNs, RNNs, and decision trees have improved detection rates, privacy, and false alarms but are also susceptible to feature imbalance and generalization capability [12]. In massive IoT scenarios, big data frameworks (like Apache Hadoop, Apache Flink, and Apache Luminous) are employed more often to deal with vast amounts of traffic with minimal latency and fault tolerance [13]. Spark, especially, has proven quite promising for real-time intrusion detection based on distributed analytics, offering classifications for various attack types [14]. Along with these methods, transformer models based on attention mechanisms have appeared for their ability to recognize far relationships within a network traffic context, and self-adaptive optimization methods (for instance, Walrus Optimization Algorithm (WOA)) assist in enhanced feature selection, anomaly detection and their effectiveness [15].

The advent of IoT and formative complex networks has placed IDS in the face of asymmetric traffic, changes in threat feature distribution, and challenges arising from the process of extracting fine- and high-level features to analyze network behaviour. Inquisitive pre-processing techniques, heterogeneous feature extraction, attention-based architectures, and self-adaptive optimization can potentially enhance detection accuracy, reduce false positives, and deliver effective intrusion detection. To address these issues, this paper proposes a feature-diversity attention classification and adaptive optimization model known as the Triplet Folded Features Attention Transformer Network (TFF_ATNet). The main contributions of this paper are as follows:

- A Triplet Fusion of Features (TFF) module that combines SESF, DLSF, and DSF to extract low-level and high-level representations and form a more balanced and discriminative feature space.
- A Trans Attention classifier utilizing transformer-based multi-head self-attention to dynamically emphasize meaningful features, enhancing precision and robustness in intrusion detection.
- The Self-Adaptive Walrus Optimization Algorithm (SA-WaOA) is used to fine-tune model hyperparameters, and optimize the learning process and improve detection performance.

The organization of this the next is the paper: Section 2 discusses relevant research on this topic; Section 3 outlines the actual proposed example; Section 4 is dedicated to comparisons of the outcomes of this model with other methods; Section 5 addresses the models' importance, implications, and future perspectives; and Section 6 concludes. this research.

2 LITERATURE REVIEW

This section reviews some of the most recent research works relating to IDS.

In recent years, there has been growing research interest in enhancing intrusion detection systems (IDS) through deep learning and hybrid approaches. *Xu et. al.,* suggested a sophisticated IDS framework that focused on noise reduction, feature extraction, temporal feature mining, and attention mechanisms as a way to decrease false positives and increase accuracy in comparison to classical models in 2023 [16]. Nonetheless, there were challenges related to training complexity and interpretability. In 2022, Imrana *et. al.,* [17] further developed this trend by proposing an IDS known as χ^2 BidLSTM, which combined statistical modeling utilizing a long short-term memory (LSTM) network that is bidirectional to enhance classification accuracy; however, this was limited by a reliance on simulated datasets, computational cost, and a lack of generalization. At the same year Mhawi *et. al.,* contributed to this line of work by applying ensemble learning with hybrid feature selection strategies to reduce dimensionality and false alarms in ID systems, but the model still experienced redundancy and error rate challenges [18]. There is also a growing interest in hybrid and scalable approaches. Chliah *et. al.,* [19] developed a hybrid ML model for network traffic, which combines supervised and unsupervised learning and implements a k-means clustering algorithm in Apache Spark. Their approach increased performance but was limited to a small number of data records, and little discussion of scalability. Liu *et. al.,* contributed to a scalable approach by developing SCADS, which is a distributed IDS for system calls stored in Google Cloud that takes advantage of Spark. While their approach improves computational efficiency in Thomson's Model for Longitudinal Group Data by allowing the use of

parallel computation, the computations were still burdensome when performing a separate analysis with a larger data set [20]. Ricky *et. al.,* provide an analysis of several traditional statistical feature selection methods (gain ratio, chi-square, etc.), which all proved to improve model accuracy. A few traditional methods reported some strengths in handling high-volume, high-dimensional, and heterogeneous data, which demanded greater flexibility in their application for a modern IDS context [21].

Recent studies have investigated innovations in deep learning and optimization-based intrusion detection systems (IDSs). Thakkar *et. al.,* [22] proposed a multi-modal dimensionality reduction algorithm that utilizes Principal Component Analysis (PCA) with autoencoders to improve the precision and F1-score of their architecture of long short-term memory (LSTM); however, they encountered challenges with computation and data dependency. In recent literature, transformer-based models have emerged as a robust option: Manocchio *et. al.,* proposed the "flowtransformer," a framework for analyzing transformer-based IDS systems; this approach successfully looked at long-term dependencies, but there was no assessment for real-world applications [23]. Additionally, Nguyen *et. al.,* [24] suggested the network of transformer attention (TAN) which was evaluated regarding attack classification on automotive CAN buses using an attention-based approach in 2023, with an issue of possible degradation on accuracy based on data representations of the original self-attention module. Along with the classification of assaults. in a network IDS setting, Saravanan *et. al.,* [25] suggested the Walrus An algorithm for optimization (WOA), as a transformer-based optimization algorithm used in wireless sensor networks to enhance coverage while obtaining positive performance; however, constrained implications concerning scalability, convergence, and adaptability.

While current IDS solutions have made significant advances in terms of detection accuracy, several issues still exist. First, many current deep learning-based IDS architectures involve high computational costs, hence limiting their usage in practical applications due to the inability to operate in real time in the IoT scenario. On the other hand, while transformer models excel in identifying long-term dependencies, they come with added computational cost, as well as high requirements for computing infrastructure during training. Moreover, while optimization-based IDS models provide better classification accuracy, they lack adaptability in heterogeneous network conditions, and their convergence cannot be guaranteed. Most existing feature extraction techniques focus on single-layer or stand-alone approaches, thereby limiting their effectiveness in learning low and high-level features of traffic simultaneously. There have been several challenges identified in some recent research that have not been adequately addressed, including the problem of traffic imbalance, inconsistencies in the distributions of data between training and test sets, interpretability, and scalability concerns. Little work has focused on developing a distributed processing model based on hybrid feature fusion and attention mechanism classification. There exists a need for a distributed IDS framework involving multi-level feature fusion, distributed preprocessing, adaptive optimization, and attention mechanism classification.

2.1 Problem Statement

The main challenges for network-based IDS are its complexity in training, data dependency, lack of interpretability, and performance trade-offs that degrade its effectiveness and efficiency. The problem here is the robustness of IDS, which is compromised by severe imbalances of categories in network traffic and distributional discrepancies between training and test sets. In order to overcome these obstacles, the suggested model will, hence, optimize the performance of IDS based on the introduction of the Apache Spark-based distributed system with triplet-folded feature learning and the implementation of an attention-based transformer network. This method focuses on reducing dataset complexity, improving scalability, enhancing intrusion detection accuracy, and overcoming the limitations of existing IDS models, such as high computational complexity, poor adaptability, limited feature representation, and reduced detection capability.

2.2 Research Objectives

The primary objective of this research is to develop an efficient and scalable intrusion detection framework for IoT network environments using hybrid feature fusion, transformer-based attention learning, distributed preprocessing, and adaptive optimization techniques. The specific objectives of this study are as follows:

1. To develop an Apache Spark-based distributed preprocessing framework for efficient handling

of large-scale and heterogeneous network traffic data.
2. To improve intrusion detection accuracy and feature representation using Triplet Feature Fusion through SESF, DLSF, and DSF feature integration.
3. To enhance classification performance and dependency learning using the Trans_Attention-based transformer classifier.
4. To optimize the performance of the intrusion detection framework using the SA-WaOA algorithm for reduced false alarm rates and computation time.
5. To evaluate the effectiveness and robustness of the proposed TFF_ATNet framework using benchmark intrusion detection datasets, including CIC-IDS2017, UNSW-NB15, and TON_IoT.

3 PROPOSED METHODOLOGY

The introduced IDS architecture, referred to as the Triplet Feature Fusion Attention Transformer Network (TFF_ATNet), has three primary phases: Data Collection and Preprocessing, Triplet Feature Fusion, and Trans_Attention Classifier-based Classification. Raw network traffic data is cleaned, normalized, and represented in the first phase, with positional encoding being added to preserve sequential relationships prior to partitioning into test, validation, and training sets. The second stage utilizes the Triplet Feature Fusion module, where Shallow Early-Stacked Feature Pooling (SESF) captures low-level patterns, Deep Learning-based Statistical Features (DLSF) learn high-level representations, and Dual Statistical Features (DSF) learned through maximum likelihood estimation capture distributional behaviour. These sets of features are averaged via Global Average Pooling and refined with attention mechanisms to obtain maximal discriminative power. The combined features are then processed by the TFF_ATNet, which integrates transformer encoder-decoder blocks with feedforward and multi-head self-attention layers through the Trans_Attention Classifier. Finally, classification results are produced by a softmax activation layer, while the Self-Adaptive Walrus Optimization Algorithm (SA-WaOA) hyperparameters are tuned to optimize the cross-entropy loss function, enabling efficient and credible IDS performance. Figure 1 displays the architecture of the suggested framework.

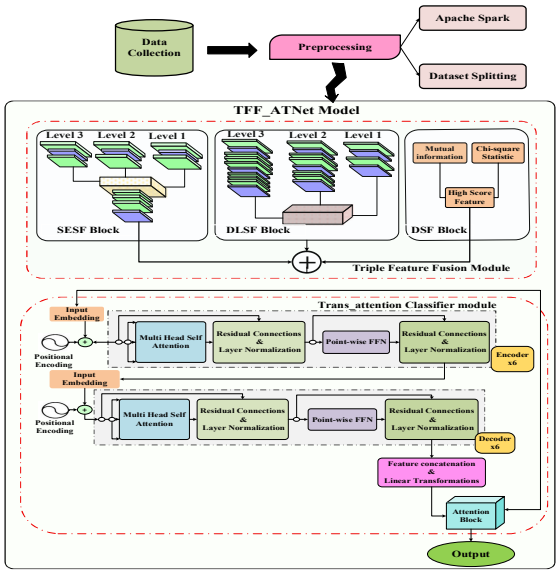


Figure 1: Overall Architecture of The Proposed Model

3.1 Preprocessing

Preprocessing is performed using Apache Spark and aggressive data partitioning to allow for effective handling of massive IDS datasets and model training with appropriately balanced models. Apache Spark provides a distributed, in-memory processing platform that supports parallel processing of the data preprocessing steps, such as cleaning, normalization, and encoding. Its execution pattern, based on DAG and data structure on RDD, increases computation by many folds, which enables the system to process massive traffic records more effectively than the traditional MapReduce methods. This supports efficient handling of noise, inconsistency, and missing values without reducing scalability for big data processing.

Data sets are subjected to systematic data splitting after pre-processing, depending on which enhances the ability of the model to generalize is enhanced. Datasets are divided into tests, validation, and training subsets in such a manner that the attack and benign instances are uniformly distributed across splits. It removes bias, prevents overfitting, and makes estimation of the proposed IDS more accurate. The use of Apache Spark-based preprocessing and partitioning of structured data allows for neat, normalized, and correctly structured input, which is used to power the following Triplet Feature Fusion step of the model.

3.2 Triplet Feature Fusion Attention Transformer Network

The TFF_ATNet extracts and concatenates a set of features in order to boost learning capabilities in the proposed system for detecting intrusions (IDS). First, Dual Data-driven Features (DSF), Deep Late-Stacked Features (DLSF), and Shallow Early-Stacked Features (SESF) are combined using Global Average Pooling (GAP) to construct a unified representation of the features. Once the features are fused, the Trans_Attention classifier is applied, consisting of a transformer with scaled dot-product attention, using residual connections and layer normalization to enable more rapid and reliable learning. Finally, fully connected layers apply activation functions and softmax for classification to facilitate more reliable and robust detection of network-based intrusions.

3.2.1 TFF module

The TFF module acts as a central entity in the suggested IDS that makes the unification of complementary features extracted from separate methodologies easier. It combines Shallow Early-Stacked Feature Pooling (SESF), which captures low-level patterns as a consequence of an earlier features fusion; Deep Late-Stacked Feature Pooling (DLSF), which gets high-level hierarchical features from deep convolutional layers; and Dual Statistical Features (DSF), to further reinforce representation using maximum likelihood estimation and statistics. The TFF module blends these three sets of features to form a robust and discriminative feature space that blends the employment of statistical descriptors and deep neural network representations to yield improved classification compared to when either source was alone in employment at the Trans Attention classification phase for efficient and accurate classification.

3.2.1.1 DSF

These further calm additional extraction of applicable information that the input can contribute toward the model, and overall improvement takes place in its learning attribute and classification capabilities. The merging of features on the high-level as well as low-level attributes, the SESF pooling mechanism makes a notable contribution towards receiving more precise classification in the Trans attention Classifier.

The maximum Likelihood method determines the likelihood of observed data under specific model parameters. For a sample $p_1, p_2, \dots p_m$, from a probability distribution with parameter(s) ϕ, where $f(p_i; \phi)$ is the probability density function, Eqn. (1) defines the maximum likelihood function,

$$L(\phi) = \prod_{i=1}^{m} f(p_i; \phi) \quad (1)$$

The chi-square statistic (Ψ_{cs}) helps to evaluate how the predicted frequency (P_n) for each class (n) corresponds to the actual observed frequency (D_n), which is important for feature selection. The formula for Ψ_{cs} is shown in the Eqn. (2) as,

$$\Psi_{cs}{}^2 = \sum_{n=1}^{l} \frac{(D_n - P_n)}{(P_n)} \quad (2)$$

Mutual information (Ψ_{mi}) indicates the amount of information received about one feature from another feature, which helps in selecting relevant features for classification. Mutual information-based feature selection is the selection of an n-feature subset from the original N-feature dataset (D) that maximizes mutual information with class labels (Q). Let R be the final selected feature subset, the mutual information can be calculated as shown in Eqn. (3),

$$\Psi_{mi} = I(R; Q) = \sum_{D_1, \dots D_n, C} P(D_1, \dots D_n, Q) \log \frac{P(D_1, \dots D_n, Q)}{P(D_1, \dots D_n, Q) P(Q)} \quad (3)$$

The high score feature vector (φ) is computed using the Eqn. (4),

$$\varphi = argmax(\Psi_{cs}, \Psi_{mi}) \quad (4)$$

After calculating the high-scoring feature vector using DSF, it is combined with other feature vectors like DLSF and SESF layer pooling; the DSF phase collaborates with DLSF and SESF pooling, which refine features using multiple convolution layers and pooling methods. This integration ensures optimal learning of characteristics at both high and low levels, which increases the classification accuracy of the model. The inclusion of statistical measures improves classification accuracy in the DSF Trans_attention Classifier.

3.2.1.2 DLSF

The proposed model of deep late stacked feature utilizes the multiple convolution and pooling layers to learn their high- and low-level features. AlexNet is very crucial in this proposed model, especially in terms of TFF learning using DLSF and SESF-based pooling. A DLSF that contains three cascaded layers as shown in Figure 3 where, *(1) Level 1*, which includes two convolutional layers with sizes of 11x11 and 5x5, respectively, followed by two max pooling layers of size 3x3. The convolution layers pick up the local patterns, whereas max pooling reduces the spatial dimension, which gives maximum emphasis on the features. *(2) Level 2* contains five convolutional layers, some of which have a filter size of 11x11. Some of them have filters of 5x5 and 3x3, while the max pooling sizes are 3x3. Additional convolution layers aid the model in understanding complexity. *(3) Level 3*, the main difference in this level is a seven convolutional layer using filters of size 11x11, 5x5, or 3x3 sizes and three max-pooling filters with 3x3 sizes. This level further refines the features learned by the previous layers.

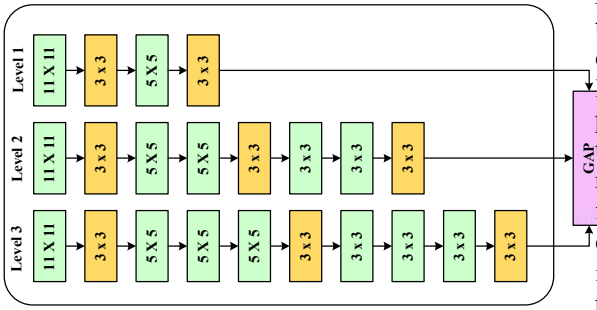


Figure 2: DLSF block in the Proposed Model

Figure 2 shows the DLSF block in the Proposed Model. Outputs from all the levels are combined with GAP before feeding into subsequent layers. GAP combines features learned at each level into a single feature representation, which aids in better integration of high-level as well as low-level features. The step retains spatial information while reducing the dimensionality. The proposed model is a deep-learning version of AlexNet. Because the DLSF takes the final stacked fusion, feature concatenation is performed from the last layer of each level, and thus, the model uses all levels' comprehensive feature information.

DLSF pooling helps to improve classification performance by stacking features from different levels to capture the intricate patterns in data. Deepening traditional AlexNet feature extraction capabilities makes it suitable for complex models. The overall robustness of the classification system is enhanced by this integration.

3.2.1.3 SESF

The shallow early-stacked feature pooling in this proposed model is characterized by an early fusion mechanism. Early integration of features at initial processing stages means that SESF pooling extracts more data in a more streamlined fashion from the input than DLSF pooling. There are three levels of the SESF pooling mechanism, each with different configurations, as shown in Figure 4. The first layer in each level employs three 11x11 convolutional layers to extract features and then uses 3x3 maximum pooling to minimize the spatial dimensions. Regarding the size of the convolutional layers, levels have a 7x7 convolutional layer in the first level, a 5x5 layer of convolution in the second level, and a 3x3 convolutional layer in the third level. These three levels of outputs are then fused using Global Average Pooling (GAP) to average the feature maps, keeping the most crucial spatial details. Then, the architecture contains three more 3x3 convolutional layers to extract further features. The last step of SESF pooling is another 3x3 max pooling, refining features to be pooled later during the stages of further processing. The early-stage mechanism for fusion is the inclusion of fusion at the front lines of the processing streams, allowing easy extraction of desirable information from the input itself. The early fusion mechanism within the SESF pooling mechanism differs from DLSF pooling because it fuses features at the initial stages of processing. These further calm additional extraction of applicable information that the input can contribute toward the model, and overall improvement takes place in its learning attribute and classification capabilities. The merging of features on the high-level as well as low-level attributes, the SESF pooling mechanism makes a notable contribution towards receiving more precise classification in the Trans attention Classifier.

Journal of Theoretical and Applied Information Technology
30st June 2026. Vol.104. No.12

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

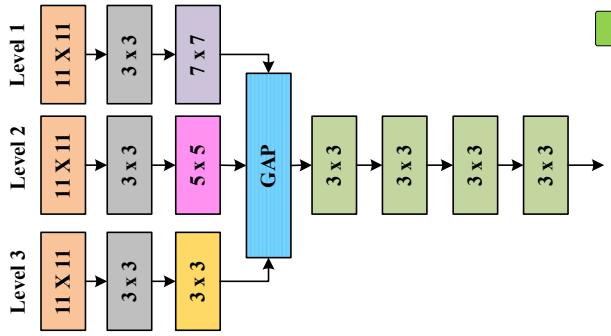


Figure 3: SLSF block in the Proposed model

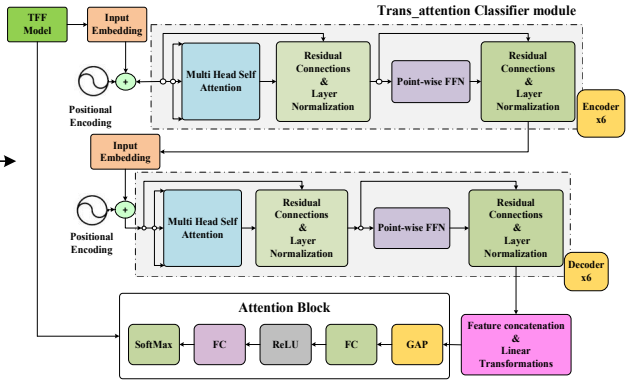


Figure 4: Trans-attention Classifier Module in the Proposed Model

The DSF computes high scores as features required for concatenation. For feature extraction and pooling, the DLSF uses a multi-layer architecture that is robust to fusion. SESF uses an early fusion mechanism with convolutional layers and max pooling. In every level, the GAP technique will be used to perform feature fusion, reducing spatial dimensions but retaining important information.

All these features are concatenated, and these final concatenated features act as an input to the Trans_attention Classifier module. The attention layer in this module, hence enables the proposed model to concentrate on pertinent characteristics for categorization purposes. The model will boost the accuracy of the classifications by combining features from many sources and using a very robust feature representation, resulting in a comprehensive feature vector.

3.2.2 Trans_attention Classifier module

The Trans_Attention Classifier outputs the combined features of the TFF module. The input features are first passed through an embedding layer and positional encoding to preserve sequential information. They are then passed through transformer blocks constituted by MHSA (multi-head self-attention) and FFN (feed-forward networks) in order to detect complex dependencies, and scaled dot-product attention helps focus on the relevant features. The TFF outputs and the transformer blocks' outputs. are merged and input into layers that are completely associated with SoftMax activation to yield ultimate categorization. Self-Adaptive Training is used to train the model. Walrus Optimization Algorithm (SA-WaOA) under A loss function that is cross-entropy with designs to prevent premature convergence and improve training stability.

3.2.2.1 Transformer Block

The transformer block in the proposed model plays a crucial role in its ability to classify input data. It has an encoder-decoder stack. For this architecture, there exist six encoders that can be further divided into an entity consisting of an MHSA network with 1024 neurons and an FFN with 1024 neurons. Some key elements inside the Transformer Block that interact together to boost the functionality of the model are as follows:

Input embedding: The input embedding block in the proposed architecture takes raw input data and converts it into sequences of vectors to supply to the models' subsequent layers. Positional encoding is designed to impose an order on the sequence by assigning an encoding for each input position, as follows: the even positional position encodings are calculated using sine functions, while those associated with odd positions are calculated via cosine functions:

For even indices,

$$EBD_{pos}(tp, 2n) = \sin(^{pos}/_{1000^{2n/d_{in}}}) \quad (5)$$

For odd indices,

$$EBD_{pos}(tp, 2n+1) = \cos(^{pos}/_{1000^{2n/d_{in}}}) \quad (6)$$

where EBD_{pos} is the positional embedding, tp is token position, n is the embedding index, and d_{in} is the embedding dimension. summation of input embeddings with positional encodings is passed to the focus on oneself layer, enabling the model to

learn both content and sequential context, thereby assisting the image feature representation and IDS performance.

Encoder and Decoder stacks: The trans-attention classifier has six encoders and decoders, each comprising an MHSA mechanism and FFN. The FFN in the encoder has 1024 neurons, and that in the decoder also includes an additional attention layer that incorporates the inputs from both the encoder and decoder. In both blocks, information flow across layers is maintained, using residual connections and layer normalization for stability and faster convergence. Information leakage has been prevented in training using a masked scaled dot-product attention (MSDPA) decoder. The output layers in the final attention layer in the decoder are compiled for final classification, which further improves the model's performance.

To process and generate sequential data, both Blocks of encoders and decoders are created in the suggested model. In each encoder block, the MHSA networks and Point-wise FFN with 1024 neurons can be found. The MHSA features of the model allow it to concentrate on various areas regarding the input sequence simultaneously. This can thus capture intricate relationships between elements. The FFN enriches the representation of the input data through an application of two linear conversions using a ReLU role of activation σ, as shown in Eqn. (7).

$$FFN(a) = \sigma(\max(0, aM_1 + v_1)\, M_2 + v_2) \quad (7)$$

The parameter 'a' represents the input vector to the FFN. $M_1$and M_2 represents the weight matrices for the initial and subsequent linear analyses in the FFN. Similarly, v_1 and v_2 represents the bias vectors for the linear transformations that are the first and second. Residuals and layer normalization are provided to ensure information flow and further stability in training. Successive encoders are computed by passing the output into the next encoder to make progressively refined learned representations for the input.

In the decoder block, the MHSA layers attend to different parts of both input and output sequences, which helps increase generative capabilities. This will avoid overfitting because, with the MSDPA mechanism, predictions for any position depend only on outputs already known at previous positions. The MSDPA can be computed using Eqn. (8) as follows,

$$Attention\ (X, Y, Z) = softmax\ (W + \frac{XY^T}{\sqrt{s_y}})Z \quad (8)$$

The term X, Y, Z denotes queries, keys, and values in Eqn. (8). The term $\sqrt{s_y}$ stabilizes gradients at the time of training. The matrix $W \in R^{k \times k}$ ensures that the model does not focus on later positions in the sequence; hence, the model prediction for the nth position only depends on known outputs up to positions less than n, which helps in decreasing overfitting.

In the self-attention layer, $-\infty$ is added to mask the SoftMax feature. The SoftMax role of activation normalizes the rows of the matrix such that they become a probability distribution. Thus, the input is represented in a new form, which is the dot product of Z with the normalized matrix. To enhance the self-attention layer, the multi-head mechanism is implemented. Each attention head has its $X/Y/Z$ weight matrix, which it calculates separately. The calculation process of each head is the same as that of the single head attention as shown in Eqn. (9) and (10).

$$MultiHead\ (X, Y, Z) = Concat\ (H_1, \ldots . H_h) M^O \quad (9)$$

$$H_n = Attention(XM_n^X, YM_n^Y, ZM_n^Z) \quad (10)$$

The results from each head are joined to create the final results. M^O is the weight matrix, H_n represents the results of the n-th attention head, each calculated independently. This method enables the model to concentrate on various positions and provide multiple representative subspaces for the attention layer.

As a final stabilization for training to converge, layer normalization '$LN(v_i)$' applies to each input sample $x \in R^s$ as in Eqn. (11),

$$LN(v_i) = \frac{v_i - \mu}{\delta} . \lambda + \omega \quad (11)$$

where v_i is the input vector, μ and δ are, respectively, the input feature's mean and standard deviation. On top of that, two trainable affine transformation parameters $\lambda \in R^s$ and $\omega \in R^s$ are

applied to the normalized input. This also ensures that the output of every layer becomes zero mean and unit variance, which makes training more stable and very likely to converge faster.

Another point-wise FFN is used in a similar way as in the encoder to enhance feature representations using two linear transformations followed by ReLU activation. Residual connections and layer normalization are also used to maintain effective flow and stability in training. After the decoder stacks have learned information, the final attention layer combines the information from both encoders and decoders to yield a complete output result for classification tasks.

Attention Block: The Attention Block utilizes the output of the multi-head self-attention (MHSA) mechanism to obtain meaningful features for classification. It performs Global Average Pooling (GAP) on the attention output, followed by two fully connected (FC) layers with ReLU activations to add non-linearity to the features. Layer normalization assists with the stability of training and improves convergence speed. The last classification is done with the SoftMax function used, which maps the weighted attention feature values to class probabilities:

$$SoftMax(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{C} e^{z_j}}, i = 1, 2, \dots, C \quad (12)$$

where z_i is the SoftMax input value for the class i and C is the quantity of classes. Moreover, masking provides a guarantee that predictions are based solely on right known outputs, thus achieving some constraints of overfitting. The feed-forward networks (FFN) within each decoder and encoder block support gradient flow when training, and this also promotes the ability of the deeper architecture to learn. The output of the SoftMax layer gives the final intrusion detection by mapping each input sample to its predicted class, and the classification task

3.3 Hyperparameters Tuning using SA-WaOA

A self-adaptive Walrus optimization algorithm is incorporated as an important module within the TFF_ATNet model for IDS in the IoT. The training models are optimized by improving learning features using triplet fusion and attention mechanisms. The quality of solutions increases with the use of a cross-entropy loss function. The random inertia weight strategy is used to prevent premature convergence and its corresponding expression is shown in Eqn. (13).

$$I_w(t) = 0.5 + \frac{c}{2} \quad (13)$$

Where $c \sim U(0,1)$

$$P_{x,y}^{S_3} = P_{x,y} + (L_{lb,y}^t + (L_{ub,y}^t - rand.L_{lb,y}^t)) \quad (14)$$

$$P_{x,y}^{S_3} = I_w.P_{x,y} + (L_{ub,y}^t - I_w.L_{ub,y}^t) \quad (15)$$

Here I_w is the weight of inertia. at time step t, 'c' Is a uniform distribution used to sample the random variable? U (0,1). '$P_{x,y}^{S_3}$' is the position of the solution in the dimension 'x' with iteration 'y' at phase 3 of 'S_3'. The local lower bound and upper bound are denoted as L_{lb} and L_{ub} respectively.

By replacing the random number with an inertia weight, this algorithm helps in achieving high-level classification performance. SA-WaoA combined with TFF_ATNet improves the accuracy of classification within the IDS framework. Optimization in an algorithm allows the model to generalize better towards unseen data, which is vital when working with dynamic IoT systems. The scaled dot-product attention mechanism enriches the additional information in the embeddings as it constructs a more informed input representation that aids classifications. This inclusion makes the model robust and efficient against all IoT security challenges.

4 RESULTS

The proposed model, implemented in Python, is evaluated using three datasets, specifically UNSW-nb15-dataset, TON_IoT datasets, and CIC-IDS2017 dataset. The proposed model's performance is compared with some prevalent techniques, such as Transformer, CNN, LSTM, RNN, and ensemble learning. The evaluation process focuses mainly on some crucial metrics like false positive rate (FPR), false negative rate (FNR), sensitivity, specificity, accuracy, and computation time. These are the metrics used in the evaluation of the performance of this model with regard to the capability to realistically detect and classify intrusions.

4.1 Data Collection

To assess the proposed opportunity detection framework, three benchmark data sets were

chosen: UNSW-NB15, TON_IoT, and CIC-IDS2017. UNSW-NB15 was created utilizing IXIA PerfectStorm and has 100 GB of network traffic that is organized into nine attack categories; it contains 49 features and consists of 175,341 records for the training materials and 82,332 records for the test collection [27]. TON_IoT provides large-scale, heterogeneous IoT/IIoT data across various networks, sensors, and logs, making it ideal for AI-based cybersecurity testing in this space [28]. CIC-IDS2017 contains real-world network traffic with attacking traffic types, including denial of service (DoS/DDoS), infiltration (to capture data), and botnets on multiple protocols, and in PCAP and CSV format [29]. Collectively, these datasets present a diverse foundation to evaluate the model in varied network environments.

4.2 Performance evaluation

The proposed TFF_ATNet framework has been tested on benchmark datasets, and its effectiveness has been contrasted with standard models (i.e., DCNN, ResNet, LSTM, Stacking Ensemble, Ensemble CNN-GRU, and Transformer) using performance metrics that included F1-score, MCC, NPV, FPR, FNR, accuracy, precision, sensitivity, and specificity as well as computation time. The results highlight that TFF_ATNet with SA-WaOA consistently outperformed all other models, with low false alarm rates, good detection accuracy, and fast computation time, proving its suitability for the identification of intrusions in real time.

Table 1 Evaluation of performance compared with proposed and existing optimization for Dataset 1

Metrics	TFF_ATNet_PSO	TFF_ATNet_GA	TFF_ATNet_WaoA	TFF_ATNet_SAWaoA
Accuracy	0.895	0.925	0.955	0.9993
Precision	0.885	0.9	0.93	0.9881
Sensitivity	0.92	0.915	0.945	0.9909
Specificity	0.89	0.935	0.91	0.9769
F1-Score	0.825	0.9	0.935	0.9882
MCC	0.855	0.9	0.96	0.98125
NPV	0.825	0.915	0.945	0.98104
FPR	0.21	0.11	0.09	0.02148
FNR	0.195	0.09	0.083	0.02077
Computation Time	330	140	110	98

Table 1 shows a comparison of TFF_ATNet with various optimization approaches, including PSO, GA, WaoA, and SA-WaOA. The SA-WaOA algorithm improved performance, resulting in the highest performance: a precision of 0.9993, an accuracy of 0.9881, a sensitivity of 0.9909, a specificity of 0.9769, and an F1-score of 0.9882. The SA-WaOA performance measures were also significantly higher than the other optimization approaches. The MCC and NPV were highest at 0.98125 and 0.98104, respectively, with the lowest rates of false positives and false negatives of 0.02148 and 0.02077. Finally, SA-WaOA had the fastest computation times (98 ms). The results indicate that SA-WaOA both improves detection performance and task efficiency relative to the other optimization approaches.

Table 2 Evaluation of performance compared with proposed and existing optimization for Dataset 2

Metrics	TFF_ATNet_PSO	TFF_ATNet_GA	TFF_ATNet_WaoA	TFF_ATNet_SAWaoA
Accuracy	0.88	0.915	0.95	0.9861
Precision	0.87	0.9	0.92	0.9794
Sensitivity	0.92	0.905	0.94	0.9921
Specificity	0.87	0.895	0.9	0.9847
F1-Score	0.81	0.9	0.925	0.97228
MCC	0.835	0.909	0.955	0.9826
NPV	0.82	0.92	0.94	0.9774
FPR	0.195	0.105	0.087	0.0219
FNR	0.18	0.085	0.092	0.02349
Computation Time	320	145	125	90

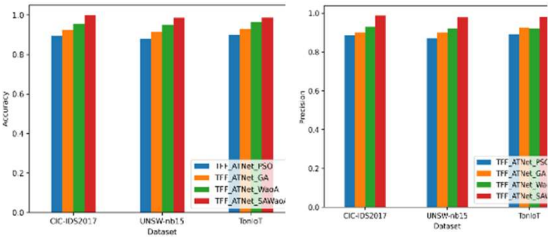
Table 2 shows how the TFF_ATNet model performed using different optimization strategies: PSO, GA, WaoA, and SA-WaOA. Overall, TFF_ATNet_SAWaoA produced better results across the evaluation metrics with precision, accuracy, sensitivity, specificity, and F1-score of 0.9861, 0.9794, 0.9921, 0.9847, and 0.97228, respectively. The MCC and NPV of the TFF_ATNet_SAWaoA were also the highest at 0.9826 and 0.9774, respectively, showing that the classifier was also reliable. This variant also recorded the lowest percentage of false positives (0.0219) and rate of false negatives (0.02349), which demonstrates error reduction. Additionally, SA-WaOA also had the least computation time of 90 ms, so it is suitable for real-time intrusion detection. The results exhibited the increased detection performance and efficient detection performance of TFF_ATNet_SAWaoA

compared with the detection performance of TFF_ATNet with PSO, GA, and WaoA.

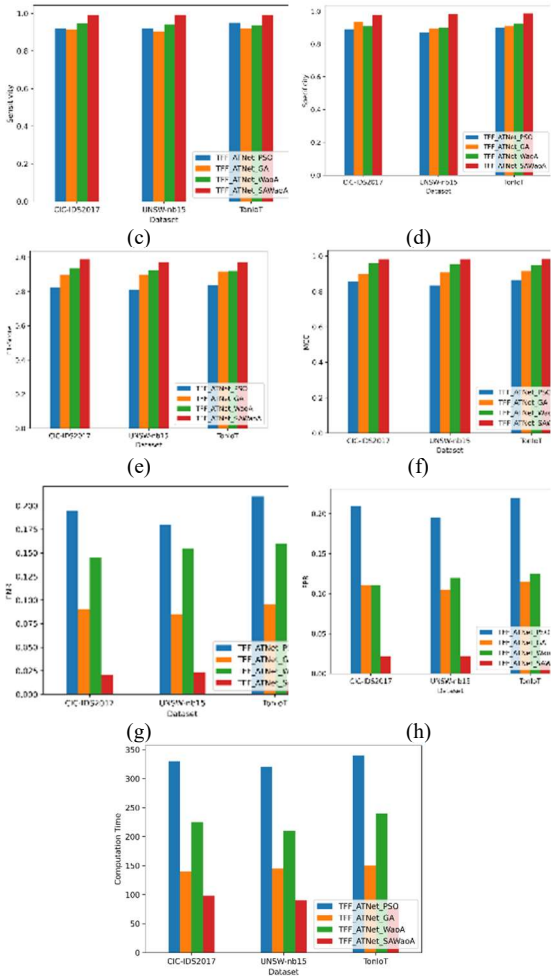
Table 3 Evaluation Of Performance Compared With Proposed And Existing Optimization For Dataset 3

Metrics	TFF_ATNet_PSO	TFF_ATNet_GA	TFF_ATNet_WaoA	TFF_ATNet_SAWaoA
Accuracy	0.9	0.93	0.965	0.987
Precision	0.89	0.925	0.92	0.9803
Sensitivity	0.95	0.92	0.935	0.9927
Specificity	0.9	0.91	0.925	0.9855
F1-Score	0.835	0.915	0.92	0.97234
MCC	0.865	0.915	0.95	0.98283
NPV	0.83	0.91	0.955	0.97196
FPR	0.22	0.115	0.087	0.0131
FNR	0.21	0.095	0.075	0.01362
Computation Time	340	150	120	78

Table 3 presents the performance of TFF_ATNet with different strategies of optimization, such as PSO, GA, WaoA, and SA-WaOA. The optimal performance was achieved in the SA-WaOA variant with an accuracy of 0.987, precision of 0.9803, sensitivity of 0.9927, specificity of 0.9855, and F1-score of 0.97234, outperforming the other options. The MCC and NPV were also higher in SA-WaOA (at 0.98283 and 0.97196, respectively), which are good indicators of reliable performance and strong classification. The model boasted the least false positive rate (0.0131) and false negative rate (0.01362), demonstrating the capability for a reduced error rate. Additionally, SA-WaOA provided the least computation time at 78 ms, indicating the proposed optimization improves the efficiency and accurate classification of TFF_ATNet for intrusion detection.



(a) (b)



(c) (d)

(e) (f)

(g) (h)

Figure 5 (A)-(I): Performance Metrics Analysis Of The Proposed Model

The proposed measures, such as specificity, sensitivity, accuracy, and precision, F1-Score, MCC, FPR, NPV, FNR, Computation Time, are visually depicted in Figures 5 (a) to (i).

Table 4 Evaluation Of Performance Compared With Proposed And Existing Methods For Dataset 1

Metrics	**DCNN**	**RESNET**	**LSTM**	**Stacking Ensemble**	**Ensemble_CNN_GRU**	**Transformer**	**Proposed**
Accuracy	0.91 24	0.9 374	0. 89 56	0.9159	0.91 83	0.94 01	0.9 993
Precision	0. 92 76	0.9 142	0. 86 97	0.9364	0.93 88	0.91 68	0.9 881

Sensitivity	0.90668	0.9385	0.9267	0.8762	0.8789	0.9398	0.9909
Specificity	0.92661	0.9056	0.88831	0.9202	0.9221	0.9085	0.9769
F1-Score	0.93554	0.92762	0.831266	0.93245	0.9347	0.9311	0.9882
MCC	0.93302	0.94448	0.838955	0.92713	0.9289	0.9456	0.98125
NPV	0.92884	0.93806	0.833077	0.93368	0.9345	0.9402	0.98104
FPR	0.0971	0.1295	0.1983	0.0955	0.1002	0.1368	0.02148
FNR	0.09312	0.1521	0.1758	0.1669	0.1726	0.1574	0.02077
Computation Time	123	210	321	256	263	205	98

Table 4 illustrates a comparison between TFF_ATNet and the base architectures being tested (DCNN, ResNet, LSTM, Stacking Ensemble, Ensemble CNN-GRU, and Transformer). The result of the proposed model was best among all models with the following measures: accuracy 0.9993, precision 0.9881, sensitivity 0.9909, specificity 0.9769, and F1-score 0.9882. The measures MCC and NPV were the best at 0.98125 and 0.98104, and the false positive and negative rates were the lowest (0.02148 and 0.02077). Furthermore, it had the fastest computation time of 98 ms, which further demonstrates that TFF_ATNet had the best accuracy and effectiveness for real-time intrusion detection.

Table 5 Evaluation Of Performance Compared With Proposed And Existing Methods For Dataset 2

Metrics	DCNN	RESNET	LSTM	Stacking Ensemble	Ensemble CNN_GRU	Transformer	Proposed
Accuracy	0.90027	0.9308	0.89117	0.9237	0.8987	0.9143	0.9861
Precision	0.93994	0.9451	0.78786	0.9203	0.8752	0.9289	0.9794
Sensitivity	0.87789	0.9198	0.91124	0.8871	0.9285	0.9076	0.9921
Specificity	0.90987	0.9312	0.89138	0.9265	0.8852	0.9285	0.9847
F1-Score	0.931665	0.92723	0.8290909	0.92521	0.8345	0.9362	0.97228
MCC	0.9202029	0.94287	0.8529898	0.93809	0.8408	0.9335	0.9826
NPV	0.933447	0.92775	0.8319199	0.92903	0.8352	0.9301	0.9774
FPR	0.125433	0.14152	0.083	0.185	0.2067	0.0996	0.0219
FNR	0.1524	0.14854	0.09532	0.1189	0.1821	0.0903	0.02349
Computation Time	109	187	143	124	318	126	90

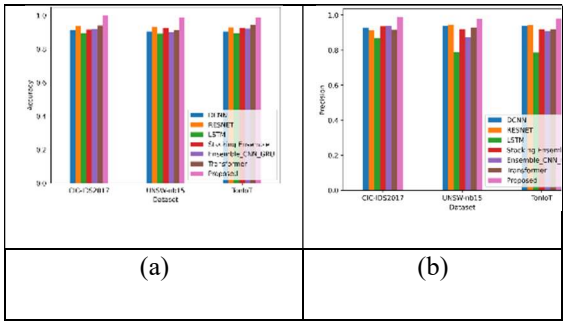
Table 5 displays a comparison of TFF_ATNet to baseline models. The proposed model attained the best performance, with the following metrics: an accuracy of 0.9861, a precision of 0.9794, a sensitivity of 0.9921, a specificity of 0.9847, and an F1-score of 0.97228. The MCC and NPV were also the highest (0.9826; 0.9774), while the false positive and false negative rates were the lowest (0.0219; 0.02349). Lastly, the proposed model also had the lowest computation time (90 ms), which shows that the proposed model outperforms the remaining models in accuracy and efficiency when it is being considered for intrusion detection.

Table 6 Evaluation Of Performance Compared With Proposed And Existing Methods For Dataset 3

Metrics	DCNN	RESNET	LSTM	Stacking Ensemble	Ensemble CNN_GRU	Transformer	Proposed
Accuracy	0.904	0.9293	0.893	0.9241	0.9213	0.9448	0.987
Precision	0.93986	0.9443	0.78556	0.919	0.9091	0.9195	0.9803
Sensitivity	0.87877	0.9205	0.914665	0.8878	0.921	0.9423	0.9927
Specificity	0.90998	0.9295	0.89545	0.9248	0.9076	0.9121	0.9855

F1-Score	0.93183	0.92697	0.82935	0.92534	0.9182	0.9354	0.97234
MCC	0.90001	0.92296	0.85253	0.93797	0.9131	0.9478	0.98283
NPV	0.91409	0.92815	0.83127	0.92921	0.9368	0.9421	0.97196
FPR	0.09507	0.1134	0.1066	0.1843	0.215	0.1442	0.0131
FNR	0.09555	0.16839	0.1561	0.1204	0.1902	0.1679	0.01362
Computation Time	109	210	189	143	314	200	78

Table 6 presents a comparison of TFF_ATNet's performance against various baseline deep learning models, namely DCNN, ResNet, LSTM, Stacking Ensemble, Ensemble CNN-GRU, and Transformer. TFF_ATNet boasts the highest performance across baseline models while providing the following benchmarks; accuracy 0.987, precision 0.9803, sensitivity 0.9927, specificity 0.9855, and F1-score 0.97234. The Matthews correlation coefficient and negative predictive value were comparatively superior at 0.98283 and 0.97196, while false positive and false negative rates were the lowest (0.0131 and 0.01362, respectively), indicating highly reliable classification. And as with previous results, the mentioned performance was achieved in the least amount of computation time (78 ms), demonstrating high accuracy and efficiency for real-time intrusion detection. These empirical findings support the position that triplet feature fusion, attention-based transformer architecture, and the SA-WaOA approach advances are greatly superior to using conventional deep learning approaches.



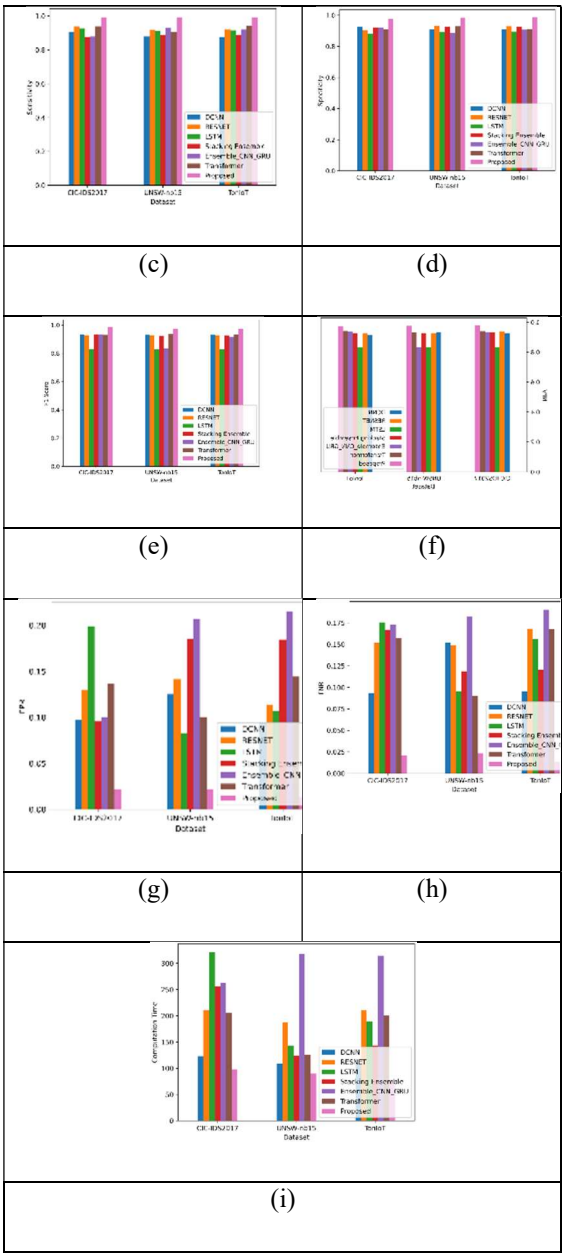


Figure 6 (A)-(I): Comparison Of The Proposed Model With Existing Models

The proposed measures, such as specificity, sensitivity, accuracy, and precision, F1-Score, MCC, NPV, FPR, FNR, Computation Time, are visually depicted in Figures 6 (a) to (i).

4.3 Tuning result

Tuning hyperparameters made a notable difference in accuracy across the evaluated datasets. The optimal accuracy (0.9919) for the CIC-IDS2017 dataset came at 100 epochs, 64 batch size, and 0.001 learning rate. The UNSW-NB15 dataset at the highest accuracy (0.9828) used the same learning

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

rate and batch size at 40 epochs, while TON_IoT achieved (0.9706) using similar settings. Overall, tuning the learning rates lower and batch sizes larger improved accuracy while demonstrating that the model is sensitive to hyperparameter tuning and needed the careful consideration of epochs to avoid overfitting or underfitting.

Table 7: Accuracy Analysis of the Proposed Model

Learning Rate	Batch Size	Epochs	Test Accuracy		
			CIC-IDS2017	UNSW-NB15	TON_IoT
0.01	16	20	0.9297	0.7727	0.8072
0.01	16	60	0.7702	0.9389	0.7525
0.01	32	20	0.9174	0.7931	0.9021
0.01	32	60	0.9024	0.7561	0.7635
0.01	32	80	0.9146	0.7814	0.8892
0.01	64	40	0.7556	0.7885	0.9245
0.01	64	80	0.9245	0.9163	0.8574
0.01	64	100	0.8316	0.7969	0.9033
0.001	16	20	0.7959	0.9104	0.7819
0.001	16	40	0.793	0.9399	0.8037
0.001	16	60	0.9009	0.7671	0.8976
0.001	16	100	0.8399	0.7929	0.901
0.001	32	20	0.894	0.9478	0.7789
0.001	32	80	0.9278	0.8741	0.8301
0.001	32	100	0.9362	0.7602	0.867
0.001	64	20	0.785	0.8476	0.9113
0.001	64	40	0.8673	0.9828	0.787
0.001	64	100	0.9919	0.8209	0.9706

Table 7 demonstrates that careful tuning of batch size, learning rate, and epochs increases the performance of the proposed model. Highest accuracies were achieved of 0.9919 (CIC-IDS2017), 0.9828 (UNSW-NB15), and 0.9706 (TON_IoT), all of which occurred at learning rates of 0.001 for larger batch sizes and extended training period. This indicates the robustness and adaptability of the model to different datasets.

4.4 Ablation study

This section describes an ablation study on three standard datasets for intrusion detection, i.e., CIC-IDS2017, UNSW-NB15, and TON_IoT, to evaluate how different configurations and feature selection techniques, together and in combination, contribute to the overall effectiveness of the TFF_ATNet prototype. The one that experiments used combinations of integration of Apache Spark, Chi-Square, Mutual Information, Dual Statistical Features, Deep Late Stacking, and Shallow Early Stacking. The results revealed that including the different available techniques greatly improved detection accuracy, precision, and overall generalization performance, compared to configurations alone. The feature selection techniques of Chi-Square and Mutual Information performed consistently well across datasets. The integration of Apache Spark and hybrid stacking techniques greatly improved learning efficiency and reduced computing. In general, the ablation study showed that the proposed enhancements improved classification performance and improved robust intrusion detection performance in diverse networks and environments.

Table 8 Ablation Study Of The Proposed Model Using CIC-IDS2017 Dataset

Model Configuration	Acc	Pre	Sen	Spe	F-Me
TFF_ATNet without Apache Spark	78.5	75.3	80.2	77.1	77.6
TFF_ATNet with Apache Spark	99.8	98.7	99	96.7	98
TFF_ATNet without Chi-Square	79.3	76.1	81.5	78	78.8
TFF_ATNet with Chi-Square	99.8	98.7	99	96.7	98
TFF_ATNet without Mutual Information	81	78.2	82.4	79.9	80.9
TFF_ATNet with Mutual Information	99.8	98.7	99	96.7	98
TFF_ATNet without Dual Statistical Feature	82.2	79.3	83.5	80.8	81.7
TFF_ATNet with Dual Statistical Feature	99.8	98.7	99	96.7	98
TFF_ATNet without Deep Late Stacking	84	81.7	85.1	82.4	83.3
TFF_ATNet with Deep Late Stacking	99.8	98.7	99	96.7	98
TFF_ATNet without Shallow Early Stacking	83.1	80.5	84.3	81.6	82.4
TFF_ATNet with Shallow Early Stacking	99.8	98.7	99	96.7	98

Table 8 presents the ablation study of TFF_ATNet on the CIC-IDS2017 dataset. Without enhancements, the model achieves moderate performance (accuracy 78.5–84%, precision 75.3–81.7%, sensitivity 80.2–85.1%, F-measure 77.6–83.3%). Integrating Apache Spark, Chi-Square, Mutual Information, Dual Statistical Features, Deep Late Stacking, and Shallow Early Stacking boosts performance substantially, achieving 99.8% accuracy, 98.7% precision, 99% sensitivity, 96.7% specificity, and 98% F-measure. These results confirm that distributed preprocessing, statistical feature fusion, and hybrid stacking are critical for robust and accurate intrusion detection.

Table 9 Ablation study of the proposed model using UNSW-NB15 Dataset

Model Configuration	Acc (%)	Pre (%)	Sen (%)	Spe (%)	F-Me (%)
TFF_ATNet without Apache Spark	76.5	73.4	78	74.1	75.7
TFF_ATNet with Apache Spark	98.61	97.94	99.21	98.46	97.72
TFF_ATNet without Chi-Square	77.8	74.6	79.3	75.8	76.2
TFF_ATNet with Chi-Square	98.61	97.94	99.21	98.46	97.72
TFF_ATNet without Mutual Information	79	75.1	80.1	76.9	77.2
TFF_ATNet with Mutual Information	98.61	97.94	99.21	98.46	97.72
TFF_ATNet without Dual Statistical Feature	80.5	76.3	81.2	78	78.1
TFF_ATNet with Dual Statistical Feature	98.61	97.94	99.21	98.46	97.72
TFF_ATNet without Deep Late Stacking	82	77.4	82.5	77.9	79.3
TFF_ATNet with Deep Late Stacking	98.61	97.94	99.21	98.46	97.72
TFF_ATNet without	81.5	76	81.7	77.2	78.8

Shallow Early Stacking						
TFF_ATNet with Shallow Early Stacking	98.61	97.94	99.21	98.46	97.72	

Table 9 presents the ablation study of TFF_ATNet on the UNSW-NB15 dataset. The model, without any enhancements, achieves decent performance (accuracy 76.5–82%, precision 73.4–77.4%, sensitivity 78–82.5%, F-measure 75.7–79.3%). By integrating Apache Spark, Chi-Square, Mutual Information, Dual Statistical Features, Deep Late Stacking, and Shallow Early Stacking, the performance is dramatically improved to an accuracy of 98.61%, precision of 97.94%, sensitivity of 99.21%, specificity of 98.46%, and F-measure of 97.72% overall. The outcomes show that high detection accuracy and stable classification on the UNSW-NB15 dataset are dependent on distributed preprocessing (Apache Spark, Chi-Square, Mutual Information), statistical features fusion (Dual Statistical Features, Deep late stacking), and hybrid stacking techniques (Shallow Early Stacking).

Table 10 Ablation study of the proposed model using TON_IoT Dataset

Model Configuration	**Acc**	**Pre**	**Sen**	**Spe**	**F-Me**
TFF_ATNet without Apache Spark	82.4	74.9	81.3	75.5	75.8
TFF_ATNet with Apache Spark	98.7	98.03	99.27	98.55	97.23
TFF_ATNet without Chi-Square	80.7	72.4	79.5	74.8	74.1
TFF_ATNet with Chi-Square	98.7	98.03	99.27	98.55	97.23
TFF_ATNet without Mutual Information	83.2	75	82.1	76.4	75.6
TFF_ATNet with Mutual Information	98.7	98.03	99.27	98.55	97.23
TFF_ATNet without Dual Statistical Feature	84.1	76.2	83.4	78.6	76.9
TFF_ATNet with Dual Statistical Feature	98.7	98.03	99.27	98.55	97.23
TFF_ATNet without Deep Late Stacking	85.3	77.5	84.8	79.9	78.1
TFF_ATNet with Deep Late Stacking	98.7	98.03	99.27	98.55	97.23
TFF_ATNet without Shallow Early Stacking	86.4	78.1	85.6	80.7	78.9
TFF_ATNet with Shallow Early Stacking	98.7	98.03	99.27	98.55	97.23

Table 10 shows an ablation experiment of TFF_ATNet on the TON_IoT dataset. In the absence of any modifications, the TFF_ATNet model shows average performance measures, notably with accuracy measuring between 82.4% and 86.4%, precision 72.4 to 78.1%, sensitivity 79.5 to 85.6% and F-measure 75.8% and 78.9% overall. However, through the incorporation of Apache Spark, Chi-Square, Mutual Information, Dual Statistical Features, Deep Late Stacking, and Shallow Early Stacking, it shows a significant performance gain with overall accuracy of 98.7%, precision of 98.03%, sensitivity of 99.27%, specificity of 98.55%, and F-measure of 97.23%. The outcomes show improvements in distributed pre-processing, statistical feature fusion, and hybrid stacking, considerably increasing the robustness, accuracy, and speed of intrusion detection designed with the TON_IoT dataset.

4.5 Discussion

According to the analysis, TFF_ATNet_WaoA is more efficient than the other models, TFF_ATNet_PSO and TFF_ATNet_GA. TFF_ATNet_SAWaoA is an improvement on TFF_ATNet_WaoA with the addition of a self-adaptive version of WaoA, which allows better adaptability to changing situations in the network as well as overall increased efficiency. This increases the ability of IDS to achieve high specificity, reduce FPRs, and minimize computations. It manages to solve the problem of imbalance within the network traffic, uses Apache Spark technology, and learns the relevant features of the data.

When comparing the proposed framework to other techniques described in the IDS literature, it

becomes evident that our approach exhibits better performance in accuracy, sensitivity, reduced computation time, and decreased false alarms. Traditional deep learning and transformer-based IDS models are usually constrained by their dependence on single-level feature extraction, thus resulting in increased computational costs and poor scalability. By leveraging the combination of SESF, DLSF, and DSF through Triplet Feature Fusion in the proposed architecture, it is possible to achieve generalized feature learning and enhanced robustness during classification across different types of datasets. Additionally, the use of distributed data preprocessing based on Apache Spark enables improved scalability and facilitates the analysis of big volumes of IoT network traffic data. Finally, by utilizing SA-WaOA optimization technique, the authors can significantly improve convergence stability and adaptive learning capabilities, yielding lower rates of false positive and false negative alerts than traditional approaches.

The model is a revolutionary approach towards cybersecurity in the field of IoT, as far as attack detection and effective classification, lowering of operational costs, and the ability to adapt to the changing networks are concerned. But, there is always scope for improvement as far as this model is concerned. The major weaknesses of this particular model include the complexity involved in training, reliance on large datasets, computation demands, and lack of transparency. Future work could be done to address all these issues. Other topics of future interest include zero-day attack detection and federated learning.

4.6 Difference from Existing Research and Research Contribution

The recent IDS approaches have made substantial improvements in classification accuracy through the application of deep learning approaches, transformer models, and optimization techniques. Nonetheless, some IDS approaches still face challenges such as scalability issues, high computational complexity, inadequate feature representation, and low adaptability for dynamic IoT systems. Hence, this chapter provides a comparative evaluation of the proposed approach with previous IDS approaches to demonstrate the significance of the proposed IDS over contemporary methods. A comparison table is provided

Table 11 Comparison Of Proposed TFF_Atnet With Existing IDS Methods

Method	Feature Extraction	Attention Mechanism	Optimization	Distributed Preprocessing	Real-Time Capability	Scalability
Traditional ML-based IDS	Single-level statistical features	Not included	Conventional optimization	Limited	Moderate	Limited
CNN/LSTM-based IDS	Deep features only	Limited attention learning	Basic optimization methods	Not supported	Moderate	Moderate
Transformer-based IDS	Transformer feature learning	Self-attention	Limited adaptive optimization	Rarely integrated	High computation overhead	Moderate
Proposed TFF_ATNet	SESF + DLSF + DSF hybrid feature fusion	Trans_Attention multi-head mechanism	SA-WaOA adaptive optimization	Apache Spark-based distributed preprocessing	High with low computation time	High

The proposed architecture is distinguished from previous IDS systems by its use of hybrid feature fusion, attention learning through transformers, distributed preprocessing, and adaptive optimization. In contrast to traditional architectures that depend on only one layer for feature extraction, the new system utilizes a combination of SESF, DLSF, and DSF to effectively learn the traffic representation. Moreover, Apache Spark offers better scalability while SA-WOAO provides greater flexibility and fast convergence. The experiments conducted show increased accuracy with reduced false alarm, false negative rates, and faster computation time than existing IDS systems.

5 CONCLUSION

This research proposed a novel intrusion detection framework, namely TFF_ATNet, to address major challenges in IDS, such as feature

imbalance, distributional inconsistencies, scalability limitations, and high false alarm rates in IoT network environments. The objectives of the study were successfully achieved through the integration of Triplet Feature Fusion using SESF, DLSF, and DSF for comprehensive traffic representation learning, Trans_Attention-based classification for effective dependency learning, SA-WaOA optimization for improved convergence and adaptive detection capability, and Apache Spark-based distributed preprocessing for scalable and efficient data handling. The integration of hybrid feature fusion significantly improved generalization capability by capturing both low-level and high-level traffic characteristics, while the SA-WaOA optimization enhanced classification stability and reduced computational overhead.

The proposed framework was evaluated using benchmark datasets including CIC-IDS2017, UNSW-NB15, and TON_IoT, achieving superior performance with 0.9993 accuracy, 0.9909 sensitivity, 0.9769 specificity, and computation time below 98 ms. The obtained results demonstrate the practical applicability of the proposed framework for IoT cybersecurity environments by improving detection capability, reducing false positive and false negative rates, and enhancing overall robustness compared with existing IDS approaches. These findings indicate that the combination of distributed preprocessing, adaptive optimization, and attention-driven hybrid feature learning provides an effective solution for next-generation intrusion detection systems.

Despite the promising performance, the proposed framework still has certain limitations, including computational complexity during training, dependency on benchmark datasets, limited interpretability of deep learning decisions, and challenges associated with real-world deployment in highly dynamic IoT infrastructures. Therefore, future work will focus on developing lightweight and explainable IDS models, integrating federated learning mechanisms for privacy-preserving distributed detection, improving zero-day attack detection capability, and enhancing model adaptability for large-scale real-world cybersecurity applications.

DECLARATIONS:

DATA AVAILABILITY STATEMENT

All the data is collected from the simulation reports of the software and tools used by the authors. Authors are working on implementing the same using real world data with appropriate permissions.

FUNDING

No funds received for this project

CONFLICTS OF INTEREST

The authors declare that they have no conflict of interest.

ETHICAL APPROVAL AND HUMAN PARTICIPATION

No ethics approval is required.

REFERENCES

[1] Amouri, A., Alaparthy, V., & Morgera, S. (2020). A Machine Learning-Based Intrusion Detection System For Mobile Internet Of Things. *Sensors (Basel, Switzerland)*, 20. Https://Doi.Org/10.3390/S20020461.

[2] Albulayhi, K., Al-Haija, Q., Alsuhibany, S., Jillepalli, A., Ashrafuzzaman, M., & Sheldon, F. (2022). Iot Intrusion Detection Using Machine Learning With A Novel High Performing Feature Selection Method. *Applied Sciences*. Https://Doi.Org/10.3390/App12105015.

[3] Zhang, Y., Yang, C., Huang, K., & Li, Y. (2023). Intrusion Detection Of Industrial Internet-Of-Things Based On Reconstructed Graph Neural Networks. IEEE Transactions On Network Science And Engineering, 10, 2894-2905. Https://Doi.Org/10.1109/TNSE.2022.3184975.

[4] Abdelmoumin, G., Rawat, D., & Rahman, A. (2021). On The Performance Of Machine Learning Models For Anomaly-Based Intelligent Intrusion Detection Systems For The Internet Of Things. *IEEE Internet Of Things Journal*, 9, 4280-4290. Https://Doi.Org/10.1109/Jiot.2021.3103829.

[5] Kimber, T., Chen, Y., & Volkamer, A. (2021). Deep Learning In Virtual Screening: Recent Applications And Developments. *International Journal Of Molecular Sciences*, 22. Https://Doi.Org/10.3390/Ijms22094435.

[6] Lei, S., Xia, C., Li, Z., Li, X., & Wang, T. (2021). HNN: A Novel Model To Study The Intrusion Detection Based On Multi-Feature Correlation And Temporal-Spatial Analysis. *IEEE Transactions On Network Science And Engineering*, 8, 3257-3274. Https://Doi.Org/10.1109/Tnse.2021.3109644.

[7] Dongen, G., & Poel, D. (2021). Influencing Factors In The Scalability Of Distributed Stream Processing Jobs. *IEEE Access*, 9, 109413-109431. Https://Doi.Org/10.1109/ACCESS.2021.3102645.

[8] Atefinia, R., & Ahmadi, M. (2022). Performance Evaluation Of Apache Spark Mllib Algorithms On An Intrusion Detection Dataset. *Arxiv*, Abs/2212.05269. Https://Doi.Org/10.22108/Jcs.2022.131400.1085.

[9] Hagar, A., & Gawali, B. (2022). Apache Spark And Deep Learning Models For High-Performance Network Intrusion Detection Using CSE-CIC-IDS2018. *Computational Intelligence And Neuroscience*, 2022. Https://Doi.Org/10.1155/2022/3131153.

[10] Tang, W., Wu, K., Zhang, Y., & Zhan, Y. (2023). A Siamese Network Based On Multiple Attention And Multilayer Transformers For Change Detection. *IEEE Transactions On Geoscience And Remote Sensing*, 61, 1-15. Https://Doi.Org/10.1109/TGRS.2023.3325220.

[11] Long, Z., Yan, H., Shen, G., Zhang, X., He, H., & Cheng, L. (2024). A Transformer-Based Network Intrusion Detection Approach For Cloud Security. *Journal Of Cloud Computing*, *13*(1), 5. Https://Doi.Org/10.1186/S13677-023-00574-9

[12] Xi, C., Wang, H., & Wang, X. (2024). A Novel Multi-Scale Network Intrusion Detection Model With Transformer. *Scientific Reports*, *14*(1). Https://Doi.Org/10.1038/S41598-024-74214-W

[13] Liang, S., Hua, Z., & Li, J. (2023). Enhanced Self-Attention Network For Remote Sensing Building Change Detection. *IEEE Journal Of Selected Topics In Applied Earth Observations And Remote Sensing*, 16, 4900-4915. Https://Doi.Org/10.1109/JSTARS.2023.3278726.

[14] Yang, Z., Liu, Z., Zong, X., & Wang, G. (2022). An Optimized Adaptive Ensemble Model With Feature Selection For Network Intrusion Detection. *Concurrency And Computation: Practice And Experience*, 35. Https://Doi.Org/10.1002/Cpe.7529.

[15] Bahmaid, S., & Ghaleb, S. A. M. (2024). Intrusion Detection System Using Chaotic Walrus Optimization-Based Convolutional Echo State Networks For Iot-Assisted Wireless Sensor Networks. *Journal Of Wireless Mobile Networks Ubiquitous Computing And Dependable Applications*, *15*(3), 236–252. Https://Doi.Org/10.58346/Jowua.2024.I3.016

[16] Geng, Y., Li, Y., & Deng, C. (2024). An Improved Binary Walrus Optimizer With Golden Sine Disturbance And Population Regeneration Mechanism To Solve Feature Selection Problems. *Biomimetics*, *9*(8), 501. Https://Doi.Org/10.3390/Biomimetics9080501

[17] Xu, H., Sun, L., Fan, G., Li, W., & Kuang, G. (2023). A Hierarchical Intrusion Detection Model Combining Multiple Deep Learning Models With Attention Mechanism. *IEEE Access*, 11, 66212-66226. Https://Doi.Org/10.1109/ACCESS.2023.3290613.

[18] Imrana, Y., Xiang, Y., Ali, L., Abdul-Rauf, Z., Hu, Y., Kadry, S., & Lim, S. (2022). X2-Bidlstm: A Feature Driven Intrusion Detection System Based On X2 Statistical Model And Bidirectional LSTM. *Sensors (Basel, Switzerland)*, 22. Https://Doi.Org/10.3390/S22052018.

[19] Mhawi, D., Aldallal, A., & Hassan, S. (2022). Advanced Feature-Selection-Based Hybrid Ensemble Learning Algorithms For Network Intrusion Detection Systems. Symmetry, 14, 1461. Https://Doi.Org/10.3390/Sym14071461.

[20] Chliah, H., Battou, A., & Laoufi, A. (2023). Hybrid Machine Learning-Based Approach For Anomaly Detection Using Apache Spark. International Journal Of Advanced Computer Science And Applications, 14(4). Https://Doi.Org/10.14569/IJACSA.2023.0140496

[21] Liu, M., Xue, Z., He, X., & Chen, J. (2021). SCADS: A Scalable Approach Using Spark In Cloud For Host-Based Intrusion Detection System With System Calls. Arxiv Preprint Arxiv:2109.11821.

[22] Ricky, Aurelius., Nurtanto, Diaz., Ketut, Gede., Darma, Putra., Naser, Jawas. (2024). 3. Comparison Of Gain Ratio And Chi-Square Feature Selection Methods In Improving SVM Performance On IDS. Lontar Komputer, 15(1). Https://Doi.Org/10.24843/Lkjiti.2024.V15.I01.P06

[23] Thakkar, A., Kikani, N., & Geddam, R. (2024). Fusion Of Linear And Non-Linear Dimensionality Reduction Techniques For Feature Reduction In LSTM-Based Intrusion Detection System. *Applied Soft Computing*, *154*, 111378. Https://Doi.Org/10.1016/J.Asoc.2024.111378

[24] Manocchio, L. D., Layeghy, S., Lo, W. W., Kulatilleke, G. K., Sarhan, M., & Portmann, M. (2024). Flowtransformer: A Transformer Framework For Flow-Based Network Intrusion Detection Systems. Expert Systems With Applications, 241, 122564. Https://Doi.Org/10.1016/J.Eswa.2023.122564

[25] Nguyen, T. P., Nam, H., & Kim, D. (2023). Transformer-Based Attention Network For In-Vehicle Intrusion Detection. IEEE Access, 11, 55389-55403. Https://Doi.Org/10.1109/ACCESS.2023.3282110

[26] Saravanan, V., Indhumathi, G., Palaniappan, R., Narayanasamy, P., Kumar, M. H., Sreekanth, K., & Navaneethan, S. (2024). A Novel Approach To Node Coverage Enhancement In Wireless Sensor Networks Using Walrus Optimization Algorithm. Results In Engineering, 103143.

[27] N. Moustafa. (2021). The UNSW-NB15 Dataset. [Online]. Available: Https://Research.Unsw.Edu.Au/Projects/Unsw-Nb15-Dataset

[28] N. Moustafa. (2021). Ton_Iot Dataset. [Online]. Available: Https://Research.Unsw.Edu.Au/Projects/Toniot-Datasets

[29] Canadian Institute For Cybersecurity. (2017). Intrusion Detection Evaluation Dataset (CIC-IDS2017). University Of New Brunswick. Https://Www.Unb.Ca/Cic/Datasets/Ids-2017.Html

ANNEXURE I

The definition for metrics used in the analysis of the proposed model is as follows.

Accuracy: It specifies the overall correctness of the model in terms of network traffic classification. High accuracy will reduce false positives and false negatives, which is important for secure networks. Accuracy is the ratio of the number of correct predictions (true positives, true negatives) divided by total predictions.

$$Accuracy = \frac{True\ Positives + True\ Negatives}{Total\ Predictions}$$

Precision: Measures the model's accuracy in identifying positive instances (threats). It's the ratio of true positive predictions to total positive predictions. High precision ensures that security personnel can trust IDS alerts, focusing efforts on genuine threats.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives}$$

Sensitivity: It is defined as the ability of the model to identify real cases correctly. The ratio of real positive predictions divided by the total number of real positive occurrences is sensitivity. High sensitivity ensures real threats are highly detected.

$$Sensitivity = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Specificity: It is a measure of the degree to which the model is able to differentiate actual negative instances from correct negatives. It can thus be defined as the ratio of true negative predictions over the total number of actual negatives. High specificity would ensure differentiation between benign and malicious traffic.

$$Specificity = \frac{True\ Negatives}{True\ Negatives + False\ Positives}$$

False Positive Rate (FPR): It refers to the measure of how likely a model is to classify benign instances wrongly as threats. Mathematically, it represents the proportion of false positive predictions to the total number of actual negative instances in the given dataset. The lower the FPR, the more reliable the model will be in terms of minimizing false alarms.

$$FPR = \frac{False\ Positives}{False\ Positives + True\ Negatives}$$

False Negative Rate (FNR): It is defined as the measure of the tendency of the model to classify actual threats as benign. Precisely, it will represent the proportion of false negative predictions and total actual positive instances in the dataset. A lower FNR will denote a more effective model in terms of the accurate detection of real threats.

$$FNR = \frac{False\ Negatives}{False\ Negatives + True\ Positives}$$

Computation time: It is the amount of time a model takes to process data and make predictions. It, therefore, quantifies the efficiency of such a model when it comes to analyzing network traffic and identifying threats. It is measured in units of milliseconds and represents all stages of the model's operation.