ISSN: 2633-4828 Vol. 5 No.4, December, 2023

International Journal of Applied Engineering & Technology

OPTIMIZATION-ENABLED DEEP LEARNING-BASED DDOS ATTACK DETECTION IN CLOUD COMPUTING

Dr. Rajbhupinder Kaur[1]
Assistant Professor in Computer Science and Engineering
Yadavindra department of engineering, Punjabi University Guru Kashi Campus, Talwandi Sabo, Bathinda, Punjab

Dr. Manoj Kumar[2]
Assistant Professor in Computer Science and Engineering
Yadavindra department of engineering, Punjabi University Guru Kashi Campus, Talwandi Sabo, Bathinda, Punjab

Corresponding Author: **Dr. Manoj Kumar**

ABSTRACT

Cloud computing is an evolving choice among businesses all over the world since it provides flexible and world wide web computer capabilities as a customizable service. Because of the dispersed nature of cloud services, security is a major problem. Traditional methods for detecting DDoS attacks struggle with scalability and real-time detection in dynamic cloud environments. This paper proposes a novel, optimization-enabled deep learning framework for detecting DDoS attacks in cloud infrastructures. The input dataset is first pre-processed using the min–max normalization technique, which replaces all of the input within a given range. The statistical features are extracted from pre-processed data. Then extracted features are forwarded to the proposed system. The proposed system consists of two stages viz. selection of features using Improved Osprey Optimization Algorithm (IOOA) and classification using Golden Gazelle Multi Neural Optimization Network (GGMNON). The GGMNON design integrates multi-layered neural network (MNN) and Golden Gazelle Hybrid Optimizer (GGHO). By leveraging GGHO algorithms to fine-tune hyperparameters, the MNN model achieves improved accuracy and efficiency in detecting anomalous traffic patterns. Extensive experimental findings show that the improved deep learning model provides a reliable way to improve cloud security against DDoS attacks, outperforming current state-of-the-art techniques in terms of detection accuracy.

Keywords: *DDoS attack, Deep learning, Osprey Optimization Algorithm, Golden Gazelle Hybrid Optimizer and Multi Layered Neural Network*

1. INTRODUCTION

Cloud computing (CC) is an Internet-enabled platform that helps businesses cut expenditures for a specific firm by providing customers or employers in large firms with access to networking, servers, databases, and other computer resources (Novaes et al., 2021; Wei et al., 2021). The preferred platform for sharing enormous data pools with a variety of features is cloud computing, which is growing in popularity. Pay-per-use cloud computing services are prevalent, whereby users are allocated to a certain pool of computers for the purpose of data mining (Elsaeidy et al., 2021; Kafhali et al., 2022). CC is one of the global commercial sectors that is growing at the quickest rate. According to Theja and Shyam (2021), Platform as a Service (PaaS), Software as a Service (SaaS), and Infrastructure as a Service (IaaS) are the three categories of services. Despite all of its advantages, security is what ultimately determines whether or not a corporation should use cloud infrastructure. In recent years, the number of DDoS assaults has enlarged due to the simplicity with which an attack may be conducted (Doshi et al., 2021; Diaz et al., 2020). Connection flooding and bandwidth flooding are the two primary types of DDoS attacks. In theory, DDoS assaults include a single server and several computers attempting to send out bogus requests at the same time (Islam *et al.,* 2021; Agrawal, & Tapaswi, 2021). The system keeps sending the same request even after it has received a response from the server (Virupakshar et al., 2020; Ray et al., 2023).

The Distributed Denial of Service (DDoS) attack is one of these serious dangers. DDoS is a lethal technique that overwhelms a host or network by flooding it with packets. By interfering with the target's services, the attack prevents authorized traffic from using them (Bhardwaj et al., 2020). Businesses or users of cloud computing only pay for service time based on length under the pay-as-you-use policy. A more serious kind of DoS attack is the DDoS assault (Aktar et al., 2023). High-level feature depictions of the traffic are extracted from low-level, granular packet data, deep learning (DL) algorithms have demonstrated outstanding performance in this context, differentiating DDoS assaults from benign traffic (Corin et al., 2020; Xu et al., 2021). A DDoS attack compromises cloud computing security and interferes with the accessibility of cloud services. The detection of DDoS assaults is necessary to ensure that services are available to authorized users. Numerous academics have researched the subject, and their findings have improved accuracy across a range of datasets.

A technique for recognizing DDoS assaults in cloud computing is presented in this article. In this study, we first examine the effects of combining deep learning with cloud computing for DDoS attack protection. We talk about the advantages of protecting against DDoS assaults as well as possible problems under this new paradigm. In this case, Golden Gazelle Multi Neural Optimization Network (GGMNON) classification using the Improved Osprey Optimization Algorithm (IOOA). MNN and GGHO are included in the GGMNON design. Through the utilization of GGHO algorithms for optimizing hyperparameters, the MNN model attains enhanced precision and efficacy in identifying aberrant traffic patterns. Feature selection, feature extraction, data augmentation, and attack detection are some of the steps involved in model detection. The objective of the study is given below,

To select features, the IOOA is introduced. It balances the phases of exploration and exploitation to find the most pertinent traits, much like ospreys do in their hunting technique. This choice streamlines the model and improves its ability to identify anomalies.

The GGMNON combines a unique Golden Gazelle Hybrid Optimizer (GGHO) that dynamically adjusts hyperparameters such learning rate with a multi-layer neural network (MNN).

The GGHO dynamically optimizes neural network hyperparameters while training by combining tactics modelled by the habits of gazelles and golden jackals. The Attention Multi layered CNN (AML-CNN) is developed to detect the attack.

The study is organized as follows: Section 1 provide the introduction of DDoS attack detection using optimization enable deep learning method. Then the literature study presented in the section 2. The research gap showed in the section 3.

2. LITERATURE REVIEW

Hussain et al. 2020 developed a unified approach that provides early recognition for DDoS attacks coordinated by a botnet that manages rogue devices using deep convolutional neural networks (CNNs). These puppet devices may create a united DDoS assault in a cell that could impede CPS operations if they each carry out quiet call, signalling, SMS spamming, or a combination of these attacks that target call, Internet, SMS, or a combination of these services, correspondingly.

Akgun *et al.,* 2022 suggested a DL model-based intrusion detection system to detect DDoS attacks and preprocessing procedures. For this purpose, a number of models based on CNN, DNN, and Long Short-Term Memory (LSTM) have been evaluated in terms of real-time and detection performance. Using the widely available CIC-DDoS2019 dataset from the literature, we tested the proposed model. We used preprocessing methods on the CIC-DDoS2019 dataset, including feature removal, random subset selection, feature selection, and normalizing. Consequently, improved recognition performance was achieved for the assessments of testing and training.

Naula, *et al.,* 2021 presented a SDN technology offers flexibility in both inline network setup and worldwide network monitoring, it has proven useful in counter-measuring complicated threats. Although there have been a number of attempts to uncover DDoS attacks, the majority of them have not used the most recent datasets, which

include the most recent threats. The creation of a modular and flexible SDN (Software Defined Networking) - based architecture was created that utilizes numerous Deep Learning (DL) and Machine Learning (ML) models to discover DDoS attacks at the application and transport layers. Through the analysis of several ML/DL methods, we were able to identify which methods are more effective in particular attack situations. The ML/DL models were investigated using the most recent security datasets, CICDoS2017 and CICDDoS2019.

Priyadarshini, & Barik, 2022 developed a special source-based DDoS security method that reduces DDoS attacks. It makes use of SDN to detect anomalous DDoS attack activity at the Network/Transport level by deploying the DDoS defender module at the SDN controller. The suggested work offers a detection technique based on DL to filter and provide while stopping malicious packets from initiating further attacks, sending valid packets to the server

Garcia, & Blandon, 2022 analysed to decrease DoS attacks. The suggested DL model is used for this classification. Dique's Graphical User Interface (GUI) allowed to monitor and read information about captured and classified packets "in real time" and to switch between the IDS and IPS modes of system operation. The CICDDoS2019 Dataset used for training and a multi-layered Deep Feed Forward neural network.

Najafimehr *et al.,* 2022 suggested a novel approach that combines unsupervised and supervised algorithms. Using a variety of flow-based criteria, a clustering algorithm first isolates the unusual traffic from the regular data. The clusters are then labeled using a categorization technique and specific numerical criteria. The suggested strategy was tested on a different set of attacks from the more recent CICDDoS2019 dataset and train on the CICIDS2017 dataset using a massive data processing framework.

Ahmim *et al.,* 2023 presented an attacker's most devastating weapons is DDoS. Classical machine learning approaches are usually inefficient and unable to manage the real properties of traffic when used to intrusion detection applications. An innovative DL-based intrusion detection system ought to be implemented at the cloud or fog level inside the framework of the IoT.. Several deep learning model types, including DNN, CNN, LSTM, and Deep Autoencoder, are combined in our hybrid model. The CIC-DDoS2019 dataset was employed to test our model.

Hasan *et al.,* 2022 suggested a hybrid intelligent system that uses DL to protect IIoT infrastructure from highly skilled and deadly multi-variant botnet attacks. The most recent dataset available, conventional and extended performance assessment measures, and the most recent DL benchmark algorithms have all been employed to thoroughly examine the suggested approach. In addition, cross-validation of our findings is carried out to distinctly display overall performance.

Aktar, & Nur, 2023 developed the effective techniques for IDS are required to thwart such attacks. The hybrid DL method was suggested to tackle these problems, which focuses on Distributed Denial of Service assaults against the Smart Grid's communication infrastructure. CNN and the Gated Recurrent Unit hybridize the suggested approach. The benchmark cyber security dataset from the Canadian Institute of Cybersecurity IDS is used for simulations. The suggested method performs better than the existing intrusion detection systems, as shown by the simulation results.

Cil, *et al.,* 2021 recommended using a DL model, like a DNN, to notice DDoS attacks on a subset of network traffic packets. The DNN model can function rapidly and precisely even with less data because it has self-updating layers and integrates feature extraction and categorization processes into its architecture. Testing was conducted using the CICDDoS2019 dataset, which includes the DDoS attack types that were created in 2019. The results demonstrated the identification of network traffic attacks.

2.1 Research gap

DL algorithms have made enormous strides in recent times to detect DDoS attacks, there remains a large gap in the literature. The majority of models in use today, such as CIC-DDoS2019, are dependent on certain datasets and may not take into consideration future developments or patterns in these types of assaults. Moreover, the concept

of hybrid models is promising overall, but there is little research on unsupervised learning, particularly when it comes to using it in conjunction with supervised methods for real-time anomaly detection. Thirdly, further research has to be done to determine how effectively these systems function in scenarios where traffic conditions are always changing and how scalable they are in dynamic environments like the Internet of Things or smart grids. Fourth, the majority of studies place more emphasis on the high accuracy of attack detection, with latency and compute efficiency to support real-time applications being considered secondary considerations. Fifth, a crucial area for comprehending the models' performance in situations with complex attacks is missing from the lack of more thorough assessments for a variety of attack vectors. By filling up the aforementioned gaps, DDoS detection systems across a range of platforms might be made more applicable and effective.

3. PROPOSED METHODOLOGY

The proposed methodology to recognize DDoS attacks in CC environments deals with applying DL optimization techniques, structured in a set of major phases that improve accuracy, efficiency, and scalability. The first step involves the preprocessing and normalization using the min-max technique; thereby all features will be on the same scale, which prevents biasing in the model. The mean, median, and standard deviation would be taken as important statistical features. The three will capture some critical patterns that are important in distinguishing normal traffic patterns from abnormal traffic. The approach employs IOOA for feature selection through appropriate balancing its exploration and exploitation stages to properly identify and refine the most characteristic ones in processing as well as reduce the degrees of complexity within the computation. Classification is implemented using a Golden Gazelle Multi Neural Optimization Network, combining a multi-layered neural network with Golden Gazelle Hybrid Optimizer GGHO for dynamic hyperparameter adaptation based on the instincts of gazelles and jackals. The novelty of this work lies in combining bio-inspired optimization with deep learning, enabling effective feature selection and dynamic neural network optimization, leading to a robust and adaptive framework for DDoS detection in the evolving landscape of cloud computing. This comprehensive approach offers a scalable, efficient, and highly accurate solution that outperforms traditional systems in speed and accuracy. Figure 1 shows the block diagram of the proposed methodology.

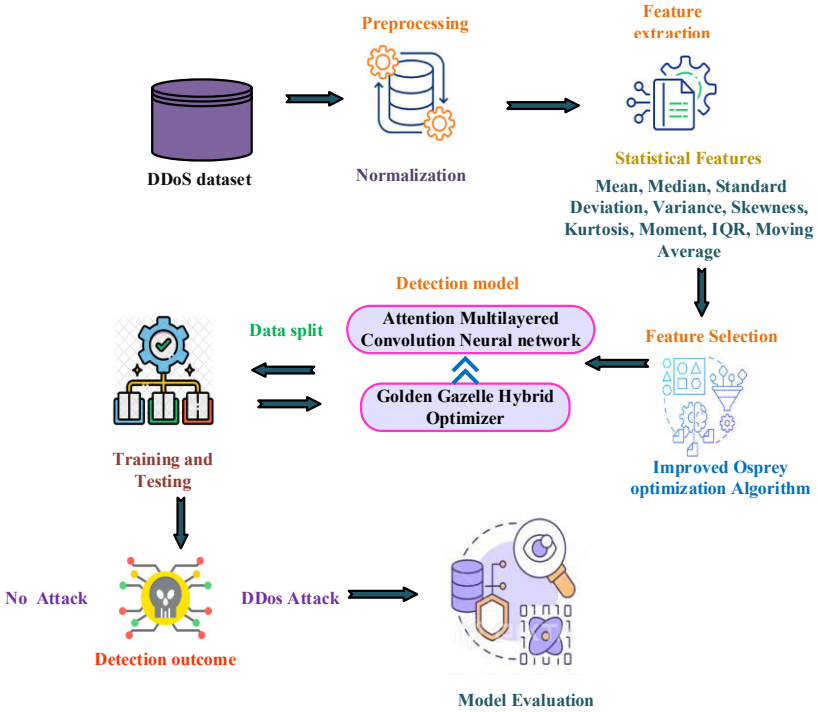

Figure 1: The proposed methodology's block diagram

3.1 Pre-processing

The first phase in the process is data preparation, which is an essential step in getting the input dataset ready for processing. In this case, all input features are scaled into a certain range using the min-max normalization technique.

3.1.1 Normalization

The min-max normalization method is employed by this system. The min-max process, which may convert the current range of data, often falling between [–1, 1] and [0, 1], is the main data scaling technique used in normalization. Equation (1) presents the normalization formula.

$$p = \frac{(x-x_{min})(max-min)}{(x_{max}-x_{min})+min} \quad (1)$$

where p is the converted input value, (min, max) is the input variable's specified range, and (x_{min}, x_{max}) is the input variable's initial range of values. This step is crucial for ensuring that the deep learning model operates effectively and efficiently without being influenced by the scale of input features. Normalized values guarantee that the data is more appropriate for the subsequent phase, which is the extraction of statistical features.

3.2 Feature Extraction

Statistical features is applied to the normalized data in order to identify key characteristics in the dataset that may be used to differentiate amid anomalous and normal network traffic.

3.2.1 Statistical features

Statistical feature extraction is applied to the normalized data to extract significant features from the dataset that may be used to differentiate between anomalous and normal network traffic. The statistical indicators provide significant information on the variability, central tendency, and distribution shape of the data. Characteristics like mean, median, and standard deviation offer a comprehensive summary of the dataset's attributes.

- ***Mean***

The arithmetic average of a distribution or set of data is called the mean. The mean is the average of the specified collection of data and is a degree of central tendency. Central tendency is the statistic that identifies the distribution of a value or a group of data. Equation 2 gives the mean formula as,

$$X' = \frac{\sum f_i}{N} \quad (2)$$

Where f_i is the frequency of each piece of data, N is the number of frequencies, and X' is the arithmetic mean.

- ***Median***

The median is the value that separates a set of data into its bigger and lesser half in the middle.

- ***Standard deviation***

The standard deviation measures the dispersion of statistical data. A statistical tool for calculating the dispersion of all the values inside a data collection with regard to the mean, or the deviation of a data value from its average, is the standard deviation. Standard deviation is given as in equation (3),

$$SD = \sqrt{\frac{N\sum X^2-(\sum X)^2}{N(N-1)}} \quad (3)$$

- ***Skewness***

The degree of asymmetry in a demand is referred to as skewness. A frequency distribution is said to be skewed to the right, or to have positive skewness, if the tail is longer to the right of the central maximum than it is to the left. The term left-skewness refers to a situation where the opposite is true, or when the skewness is negative. Skewness is a number without dimensions. Skewness formula is defined in equation (4),

$$Skewness = \frac{\sum_{i=1}^{N}(X - X')^3}{(N-1)SD^3} \quad (4)$$

where N is the total number of observations, SD is the standard deviation, X' is the sample mean.

- ***Variance***

Variance measures how far a set of data points diverges from the mean. A set of data's variances, in Layman's definition, is the degree to which the data are dispersed from the mean (average) value. Finding a variation is the expected difference between the actual value and the deviation. As a result, the standard deviation of the data set affects variance. Depends on the variance value, the data are more or less dispersed from the mean; less dispersed data are those where the variance is low or negligible. It is therefore referred to as a data spread from mean measure. The variance formula is given below

$$S^2 = \frac{\sum(x_i - \bar{x})^2}{n-1} \quad (5)$$

S^2- Sample variance

Where x_i- First observation value, $\bar{x}$- overall observation's mean value and n is the total number of observations

3.3 Inter Quartile Range (IQR)

As a statistical measure of dispersion, the interquartile range (IQR) is computed as the difference between a dataset's first (Q1) and third (Q3) quartiles. It calculates the middle 50% of the data's distribution.

$$IQR = Q3 - Q1 \quad (6)$$

Where $Q1$, is the 25th percentile of the data, or the value at which 25% of the data falls and the value below which 75% of the data falls, is represented by the third quartile, or $Q3$.

3.3.1 Moments

Moments are employed in statistics to characterize a distribution's form. The distribution's shape can be determined by taking the n^{th} moment of a data set about a point c, which is usually the mean. Calculating the k^{th} central moment is as follows:

$$\mu_k = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})^k \quad (7)$$

The k^{th} moment around the mean is denoted by μ_k, where N is the number of data points, x_i is the mean of the data points, $\bar{x}$ is the individual data point, and k is the moment's order.

3.3.2 Moving average

A moving average is a computation that averages many subsets of the entire data set and is used to study individual data points. To display patterns over time, it evens out fluctuations. The following formula (8) is employed to find a time series' simple moving average (SMA):

$$SMA = \frac{1}{n}\sum_{i=t-n+1}^{t} x_i \quad (8)$$

The current time point is t, the number of time periods is n, and the data points within the time window are x_i.

3.4 Improved Osprey for Feature selection

Feature selection, which comes after statistical feature extraction, is to select only the most pertinent features to diminish computational intricacy and enhance model performance. The proposed framework utilizes an Improved Osprey Optimization Algorithm, an optimization technique that is inspired by population-based optimization techniques characteristic of the hunting behavior of birds of prey known for systematic and swift attacks on fish by the said birds. IOOA imitates strategic hunting approaches using two phases: exploration and exploitation. An

excellent example of a natural behavior that might serve as the foundation for the development of a new optimization algorithm is the osprey's strategy of attacking fish and guiding them to a good location for feeding. The update process for the Osprey position in the two stages of exploration and exploitation is also fully described in a simulation of natural osprey activity after the OOA.

3.4.1 Initialization

The OOA is a population-based approach that relies on the search skills of the population members in the problem-solving area to identify a feasible solution through repetition. The search space in which an osprey is located specifies the values of the problem variables for any osprey in the OOA population. It is a vector that can be quantitatively expressed as a potential solution to the issue. Therefore, Eqn. (9) defines the OOA population, showing all ospreys, which can be represented as a matrix. Eqn. (10) is used to set the osprey locations within the search space randomly at the start of the OOA implementation.

$$I = \begin{bmatrix} I_1 \\ \vdots \\ I_a \\ \vdots \\ I_K \end{bmatrix}_{K\times x} = \begin{bmatrix} i_{1,1} & \cdots & i_{1,b} & \cdots & i_{1,x} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ i_{a,1} & \cdots & i_{a,b} & \cdots & i_{a,x} \\ \vdots & \therefore & \vdots & \ddots & \vdots \\ i_{K,1} & \cdots & i_{K,b} & \cdots & i_{K,x} \end{bmatrix}_{K\times x} \quad (9)$$

$$i_{a,b} = \overline{LB}_b + R_{a,b}.(\overline{UB}_b - \overline{LB}_b), a = 1,2,\ldots.,K, b = 1,2,\ldots,x, \quad (10)$$

where K is the number of ospreys, I is the population matrix of the osprey locations $I_a$is the b^{th} osprey, x is the number of problem variables, $R_{a,b}$ is a arbitrary number in the interval [0, 1], $i_{a,b}$shows its b^{th}dimension (problem variable) and $\overline{LB}_b$and $\overline{UB}_b$symbolize the lower and upper bounds.

Since every osprey is a possible answer to the issue that agrees to that specific osprey, the objective function can be evaluated. Equation (11) states that the assessed values for the problem's objective function can be signified by a vector.

$$P = \begin{bmatrix} P_1 \\ \vdots \\ P_a \\ \vdots \\ P_K \end{bmatrix}_{K\times 1} = \begin{bmatrix} P(I_1) \\ \vdots \\ P(I_a) \\ \vdots \\ P(I_K) \end{bmatrix}_{K\times 1} \quad (11)$$

where P is the vector containing the objective function values and P_1 is the computed objective function value for the a^{th} osprey.

Phase 1: Locating Positions and Evading Energy-Based Fish Hunting (Exploration)

Ospreys can find fish underwater because they are strong hunters with excellent vision. They locate the fish, hit it, and then go under the surface to catch it. A simulation of ospreys' real natural behaviour was employed to model the first step of the OOA's population update. Modelling the osprey strike on fish resulted in a considerable alteration in the osprey's position in the search area, which enhanced the OOA's capacity to search and locate the ideal location while evading local optima. The OOA concept classified each osprey as an undersea fish based on its location within the search region with a higher objective function. Each osprey's set of fish was calculated using equation (12) as follows:

$$P(f_a) = [I_m | m \in \{1,2,\ldots,K\} \wedge P_m < P_a] \cup \{I_{Bst}\} \quad (12)$$

where I_{Bst} is the optimal osprey solution and $P(f_a)$ is the position set for the a^{th} osprey. At random, the osprey finds one of these fish and strikes it. At random, the osprey finds one of these fish and attacks it. But occasionally, the prey is difficult to locate and can become lost quickly. Evading Energy E_v is the power of the prey energy. The prey still has sufficient energy to escape when $|E_v|$ is bigger than 1. Equation (13) was utilized to ascertain a new location for the corresponding osprey, taking into account the osprey's simulated flight near the fish with evading energy.

$$z_{i,j}^{P_1} = z_{i,j} + r_{i,j}.\left(SF_{i,j} - I_{i,j}.z_{i,j}\right) * E_v \quad (13)$$

$$E_v = E_{ld} \times E_0 \quad (14)$$

$$E_{ld} = 1.5 \times \left(1 - \frac{t}{T}\right) \quad (15)$$

$$E_0 = 2 \times rand - 1 \quad (16)$$

The move the osprey are regulated by the prey Evading Energy which is defined in (14). E_{ld} demonstrates that during the generation, the prey decrement energy falls linearly from 1.5 to zero. E_0, on the other hand, represents the starting energy of the prey with $rand$ being within [0, 1]. Equation (17) declares that the osprey's previous location is replaced if the objective function's value rises at the new location.

$$z_{i,j}^{P_1} = \begin{cases} z_{i,j}^{P_1}, & lb_j \le z_{i,j}^{P_1} \le ub_j \\ lb_j, & z_{i,j}^{P_1} < lb_j \\ ub_j, & z_{i,j}^{P_1} > ub_j \end{cases} \quad (17)$$

$$Z_i = \begin{cases} Z_i^{P_1}, & F_i^{P_1} < F_i \\ Z_i, & else \end{cases} \quad (18)$$

where $I_{i,j}$ are arbitrary numbers from the set $\{1,2\}$, $Z_i^{P_1}$ is the new location of the i^{th} osprey. Furthermore, SF_i is the fish that has been chosen for the i^{th} osprey, $SF_{i,j}$is its jth dimension, and $F_i^{P_1}$ is the objective function value.

Phase 2: Moving the fish to the appropriate location (exploitation)

The osprey carries the fish it has captured to a safe place to eat it. The second step of OOA's population update model is based on modeling the natural behavior of this osprey. The modelling of moving the prey to the optimal spot results in a modest variation in the osprey's position in the search area. The ability of the OOA to benefit from local search and convergence toward better solutions close to the solutions found so rises. OOA's design first applies (19) to select a new random location for each member of the population that is considered "suitable for eating fish" to mimic the ospreys' natural behaviour. If the value of the goal function is enhanced in this new location, it then replaces the previous placement of the relevant osprey in accordance with equation (21).

$$z_{i,j}^{P_2} = z_{i,j} + \left(\frac{lb_j + r\left(ub_j - lb_j\right)}{t}\right), \quad t = 1,2, \ldots., T \quad (19)$$

$$z_{i,j}^{P_2} = \begin{cases} z_{i,j}^{P_2}, & lb_j \le z_{i,j}^{P_2} \le ub_j \\ lb_j, & z_{i,j}^{P_2} < lb_j \\ ub_j, & z_{i,j}^{P_2} > ub_j \end{cases} \quad (20)$$

$$Z_i = \begin{cases} Z_i^{P_2}, & F_i^{P_2} < F_i \\ Z_i, & else \end{cases} \quad (21)$$

where $F_i^{P_2}$ is its objective function, $z_{i,j}^{P_2}$ is its j th dimension, and $Z_i^{P_2}$ is the new location of the i th osprey. T is the total number of iterations, t is the algorithm's iteration counter. By reducing the dimensionality of the dataset, the IOOA feature selection speeds up and improves the accuracy and speed of the model's overall performance during training. The flowchart of IOOA algorithm is shown in the Figure 2.

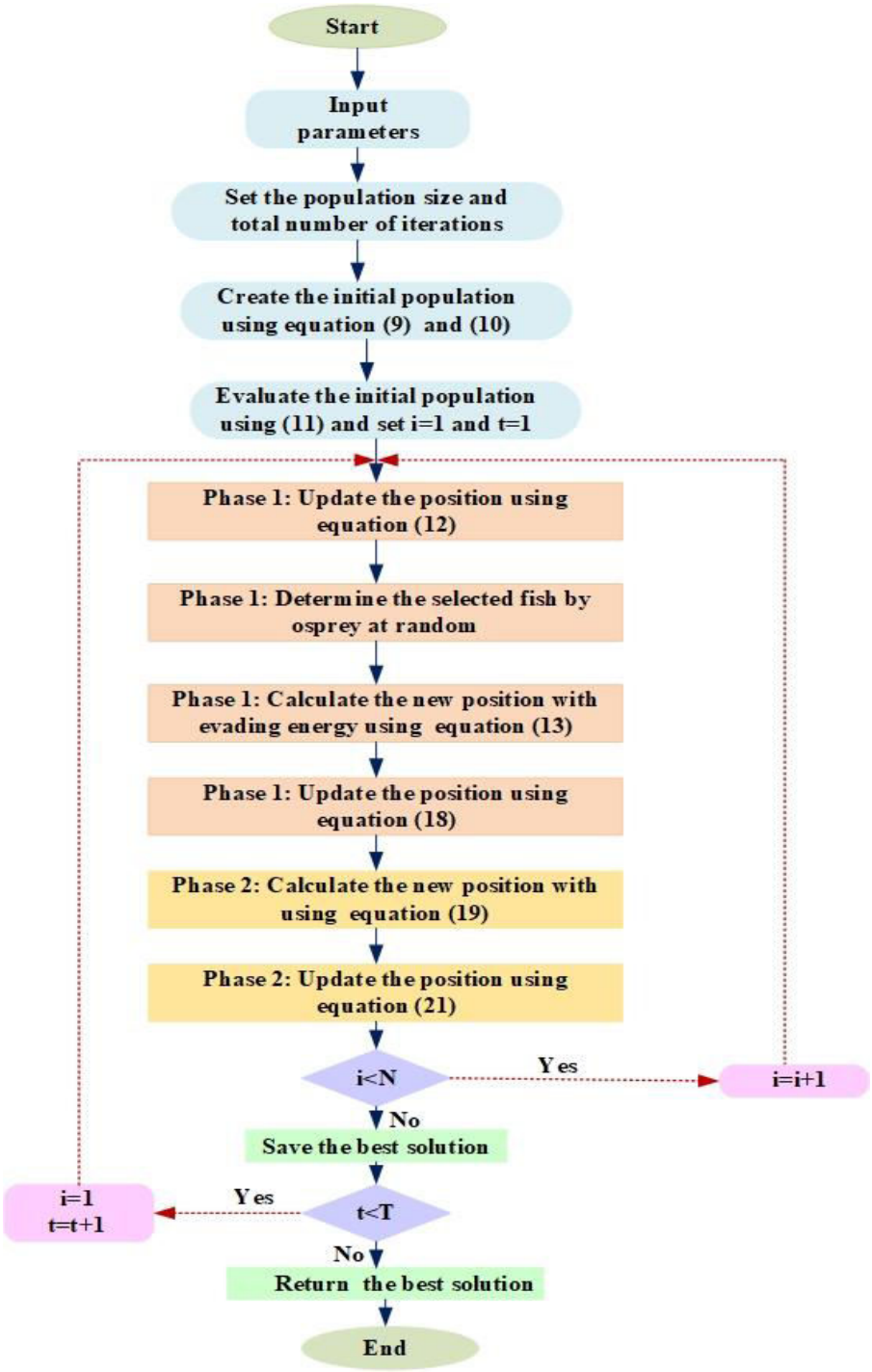

Figure 2: Flowchart of IOOA algorithm

3.4.2 Golden Gazelle Multi Neural Optimization Network

The classification phase uses a newly introduced deep learning approach known as the Golden Gazelle Multi Neural Optimization Network (GGMNON). The model is inspired by the hybrid optimization of golden jackal and gazelles, where there is a combination of a multi-layer neural network with a combination of the GGHO, which modifies the hyperparameters of the neural network. The GGHO takes its inspirations from how gazelles and jackals are hunted, in this case, through foraging and evading a predator. The GGMNON framework, indeed, is optimized so that important attributes of the neural network such as the learning rates are improved. With fine-tuning of these parameters by the GGHO, the performance of the neural network becomes more accurate and efficient while detecting any anomalous traffic pattern, indicating the DDoS attacks.

3.4.3 Attention Multi layered Convolutional Neural Network

Pooling, weight-sharing, and local receptive fields are the three concepts used by Convolutional Neural Networks. It may consist of several layers, such as FC, pooling, and convolutional layers. The essential component of the architecture is the convolution Layer, which carries out the shift, multiply, and sum operations. The major goal is to identify patterns that are consistent across the data set in small portions of the input images. The weighted total is then mixed with a bias and passed through an activation function to produce non-linearity. The convolution filters are determined by the weights of each convolutional layer; a convolutional layer may contain many filters. Figure 3 depicts the architecture of Attention Multi layered CNN.

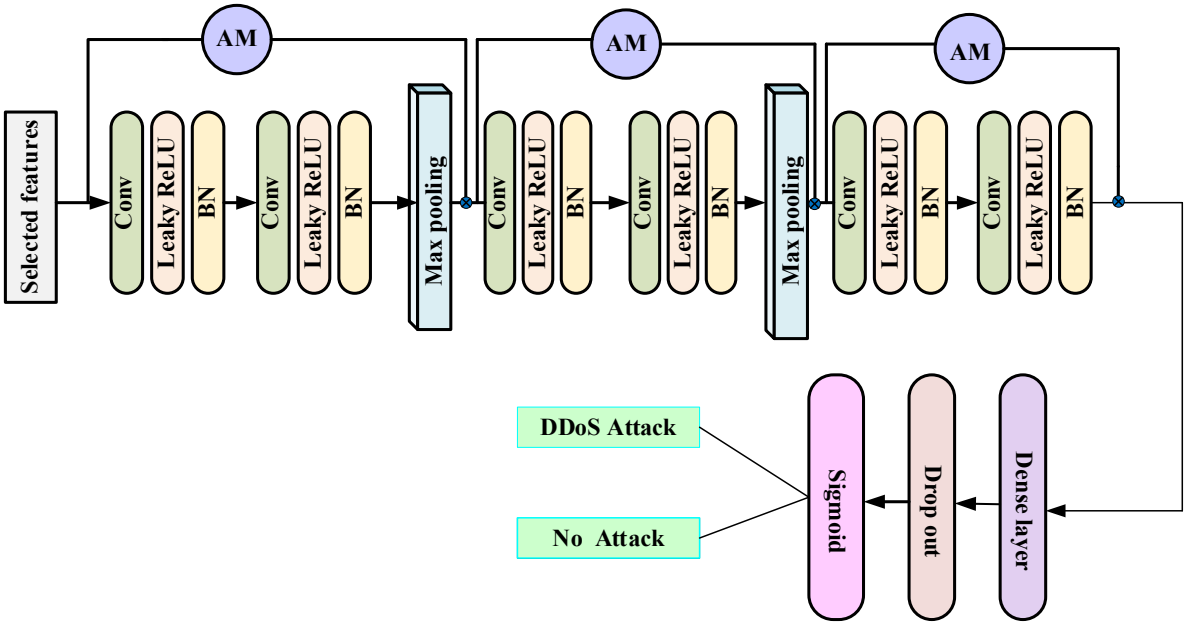

Figure 3: Architecture of Attention Multi layered Convolutional Neural Network

Every convolution layer has an optional pooling layer that comes after it to simplify the data in the output from the convolutional layer. By down sampling each feature map generated by the convolutional, a pooling layer summarizes a cluster of neurons in the convolution layer. In the proposed model, the attention module is connected between the convolution blocks and the pooling layer which is not in a traditional CNN model and is employed to extract high and low-level features for the accurate detection. In the typical CNN design, fully linked layers come after convolutional layers. The ReLU helps to partially alleviate the computational calculation problems in deep learning. The ReLU has a drawback in that it is insensitive to non-positive inputs, which causes many neurons to become inactivated during training. This may be interpreted as a regression gradient problem with negative values once more. As mentioned in Equation (22) the Leaky rectified linear unit (Leaky ReLU), which slightly activates for negative values, is added to resolve the non-activation for non-positive integers.

$$y = \begin{cases} \alpha * x & if\ x < 0,\ \ \alpha = 0.25 \\ x & if\ x \geq 0 \end{cases} \quad (22)$$

Instead of a ReLU with a horizontal slant, the one proposed here uses the LeakyRelu activation function, which is based on or an updated version of the ReLU and has minimal negative slopes for non-positive values. Dropout helps with overfitting problems and allows for the quick development of many different neural network topologies. A conventional neural network's "dropout" refers to regions that are no longer active (both apparent and hidden). Dropout has been incorporated into the model to improve the suggested architecture's performance because it inhibits overfitting, encourages the network to learn a variety of features. Experiments on the amount of dropout λ are conducted. Hyper-parameters in deep learning are external setup settings that cannot be determined from the experimentation data. These parameters, which were set before the model was trained, have a significant influence on the model's performance and learning process. The learning rate (η) is adjusted to aid in the model's convergence during training, guaranteeing the model's stability and generalization. In contrast to shallow architectures, the proposed model experiments with reduced learning rates.

3.4.4 Golden Gazelle Hybrid Optimizer

During training, the Golden Gazelle Hybrid Optimizer (GGHO) dynamically optimizes the hyperparameters of the neural network. This optimizer combines tactics derived from jackal and gazelle hunting and survival methods, modifying important parameters such as layer depth, and learning rate to boost the overall accuracy and convergence rate of the model. GGHO makes use of the habits of mountain gazelles and golden jackals, two animals renowned for having highly developed social, hunting, and survival skills. These tactics are represented mathematically to dynamically fine-tune the hyperparameters of the neural network during training, hence increasing convergence rates and accuracy.

Golden jackals use their collective cry to let others know where they are and to communicate with one other in order to find their prey. For foraging, golden jackals adopt cooperative foraging, which permits them to search an available territory of larger prey. They will circle the target to make sure it is ready for their attack, and then they will encircle it until it is unable to flee. Finally, if escape looks hopeless, they will attack the target. The hierarchical and societal requirements that gazelles display is incorporated into the mathematical formulation of the MGO. To search for food sources, this optimizer uses four main strategies: emigration, maternity herds, lone territorial males, and bachelor male herds. Every gazelle has the capacity to become a member of any of the three herds listed earlier. The presence of a mature male gazelle within the herd's region is the ideal global solution.

3.5. Territorial solitary males (MTS)

Mature male gazelles create isolated areas and are fiercely protective of them. They fight over territory or control over females. The following is a description of their territory's model:

$$M_{TS} = M_g - \left|\left(R_{i,1}.H_Y - R_{i,2}.X^t\right).f\right|.C_R \quad (23)$$

where M_g denotes the location vector of the best global solution, whereas $R_{i,1}$ and $R_{i,2}$ denote arbitrary integers either 1 or 2. The coefficient vector for the young male herd, as determined by Equation (24), is represented by the H_Y vector. Furthermore, Equations (25) and (26) is used to compute the value of f, and C_R is a coefficient vector that is determined by random selection and updated after each iteration.

$$H_Y = X_{Ra}.R_1 + M_{pr}.R_2,\ \ Ra = \{[N/3, N]\} \quad (24)$$

where M_{pr} is the average population, which is generated randomly $(N = 3)$, and X_{Ra} denotes a randomly picked solution from the range of Ra. N here denotes the size of the population. Additionally, R_1 and R_2 represent two random numbers that are in the interval [0, 1].

$$f = N_1^D.\exp\left(2 - t.\left(\frac{2}{T}\right)\right) \quad (25)$$

where a random integer drawn from a standard distribution is denoted by N_1^D. T denotes the present iteration, and t stands for the maximum number of iterations.

$$C_R = \begin{cases} (A+1) + R_3 \\ A.N_2^D \\ R_4^D \\ N_3^D.{N_4^D}^2.\cos\left((R_4.2).N_3^D\right) \end{cases} \quad (26)$$

$$A = -1 + t.\left(\frac{-1}{T}\right) \quad (27)$$

N_2, N_3, and N_4 are random numbers produced from a normal distribution, whereas R_3 and R_4 are random values selected from the interval [0, 1].

3.5.1 Maternity herds (HM)

Maternal herds are linked to the offspring of healthy male gazelles. The following can be used to model and depict this behaviour:

$$H_M = \left(H_Y + C_{2,r}\right) + \left(R_{i,3}.M_g - R_{i,4}.X_R\right).C_{3,R} \quad (28)$$

where X_R represents the position vector that was chosen at random from the population, and $R_{i,3}$ and $R_{i,4}$ stand for arbitrary integers, either 1 or 2.

3.5.2 Bachelor male herds (HBM)

Male gazelles actively seek for females as they get older and territory. During this hunt, young male gazelles involve in combat with other males to gain power over the territory and females. This action can be modelled and represented as follows:

$$H_{BM} = (X^t - D) + \left(R_{i,5}.M_g - R_{i,6}H_Y\right).C_R \quad (29)$$

$$D = \left(|X^t| + |M_g|\right).(2.R_6 - 1) \quad (30)$$

R_6 is a uniform random number selected from the interval [0, 1], while $R_{i,5}$ and $R_{i,6}$ are arbitrary integers that can have values of 2 or 1.

3.5.3 Levy-based migration in search of food and exploration

Mountain gazelles are migratory animals that use their foraging skills to travel great distances in search of food. They also have exceptional jumping ability and a terrific running speed.

This activity can be modelled and represented as:

$$E_{EF} = (UB - LB).R_7 + LB * 0.05 \times LF_D(\beta) \quad (31)$$

The problem's upper and lower bounds are signified by UB and LB, correspondingly. Furthermore, R_7 is a random number with a range of 0 to 1. Equation (32) indicates that Levy flights from Golden jackal optimization, where u_D and v_D are random integers obtained from a normal distribution, with u_D's standard deviation being σ and v_D's being 1. The parameter β of Levy flights is 1.5. The Levy flights vector's dimension is denoted by D.

$$LF_D(\beta) = \frac{u_D.\ \sigma}{|v_D|^{\frac{1}{\beta}}} \quad (32)$$

$$\sigma = \left(\frac{\Gamma(1+\beta) \times sin\left(\frac{\pi\beta}{2}\right)}{\Gamma(1+\beta) \times \beta \times 2^{\left(\frac{\beta-1}{2}\right)}} \right)^{\frac{1}{\beta}} \quad (33)$$

The four processes M_{TS}, H_M, H_{BM}, and E_{EF} are employed in turn to generate new mountain gazelle generations. Based on their performance, all-mountain gazelles are grouped in a mounting pattern at the conclusion of each era. The capacity of GGHO to dynamically adjust the hyperparameters of the neural network during training is one of its major accomplishments. GGHO continuously modifies these settings according to the search process using its bio-inspired techniques. Through dynamic adjustment, overfitting is avoided, convergence is accelerated, and the neural network's adaptability is maintained during the training process. Adjusting the learning rate during exploration and decreasing it during exploitation can help keep the model from being trapped in local minima. The pseudocode of GGHO is illustrated in the algorithm 1.

Algorithm 1: Pseudocode of GGHO

Initialize Parameters:

Define population size (N), number of iterations (T), and problem bounds (UB, LB).

For each gazelle (solution) in population:

Initialize position

Evaluate fitness of each gazelle (solution)

Identify the best global solution M_g

For each iteration t = 1 to T:

Update Territorial Solitary Males ($\boldsymbol{M_{TS}}$*)*

For each gazelle (i = 1 to N):

Compute position of solitary males using:

$$M_{TS} = M_g - \left|\left(R_{i,1}.H_Y - R_{i,2}.X^t\right).f\right|.C_R$$

Update coefficient vectors H_Y *and* C_R *using Equations (24) and (26).*

Evaluate fitness of new position M_{TS}.

Update Maternity Herds ($\boldsymbol{H_M}$*)*

For each gazelle:

Compute the position of maternity herds using:

$$H_M = \left(H_Y + C_{2,r}\right) + \left(R_{i,3}.M_g - R_{i,4}.X_R\right).C_{3,R}$$

Update $C_{2,r}$ *and other random variables.*

Evaluate fitness of the herd positions H_M.

Update Bachelor Male Herds ($\boldsymbol{H_{BM}}$*)*

For each gazelle:

Compute the position of bachelor male herds using:

$$D = \left(|X^t| + |M_g|\right).(2.R_6 - 1)$$

$$H_{BM} = (X^t - D) + \left(R_{i,5}.M_g - R_{i,6}H_Y\right).C_R$$

Evaluate the fitness of the new positions H_{BM}

Levy-based Emigration ($\boldsymbol{E_{EF}}$*)*

For each gazelle:

Perform Levy flight:

$$E_{EF} = (UB - LB).R_7 + LB * 0.05 \times LF_D(\beta)$$

Update $LF_D(\beta)$ *using Equations (32) and (33).*

Evaluate fitness of the new emigration positions E_{EF}

Update Global *Best Solution*

Evaluate fitness of updated gazelle positions

Sort population based on fitness

Update best global solution M_g

Return best solution M_g *and final optimized hyperparameters*

4. RESULT AND ANALYSIS

This section compares the suggested model's efficiency with methods that are currently in use, and the results are displayed as graphs. The proposed model was implemented using MATLAB. The data for evaluation employed in this work is the CIC-DDoS2019 Dataset (https://www.unb.ca/cic/datasets/ddos-2019.html). Numerous network traffic metrics are gathered for this data collection, such as protocol type, packet size, inter-arrival times, and even flow-based statistical variables like byte counts and packet rates. Many attack simulations, including those that use SYN, ICMP, DNS, and UDP, differ greatly from one another and can represent a variety of DDoS attack scenarios. Variants can help assess how resistant the detection systems are to different kinds of attacks. Comparative analysis involves benchmarking against established techniques Proposed, CNN, ANN, and FFNN using metrics encompassing F-measure, Negative Predictive Value (NPV), False Positive Rate (FPR), False Negative Rate (FNR), Sensitivity, Specificity, Accuracy, Precision, Recall, and Matthew's Correlation Coefficient (MCC) to robustly assess the proposed model's performance.

4.1 Performance metrics

This section contains the performance metrics and the formulae for calculating them.

4.1.1 Accuracy

The percentage of correctly classified samples relative to all samples is known as accuracy.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (34)$$

4.1.2 Precision

Precision indicates the proportion of accurately anticipated positive observations to all anticipated positive observations. The ability of the model to classify negative samples as positive is evaluated.

$$Precision = \frac{TP}{TP+FP} \quad (35)$$

4.1.3 Sensitivity

The percentage of real positive samples that were accurately predicted to be positive is known as sensitivity.

$$Sensitivity = \frac{TP}{TP+TN} \quad (36)$$

4.1.4 Specificity

The percentage of real negative samples that were correctly predicted to be negative is known as specificity. It evaluates the model's capacity to recognize every negative sample.$Specificity = \frac{TN}{TN+FP}$ (37)

4.1.5 F-Measure:

The harmonic mean of recall and precision that yields a single score that equalizes the two metrics. It illustrates how accurately the model captures both positive and negative data.

$$F - Measure = \frac{2*precision*recall}{Precision+recall} \quad (38)$$

4.1.6 FPR

FPR shows the percentage of true negative samples that were incorrectly identified as positive. It assesses how likely the model is to identify negative samples as positive.

$$FPR = \frac{FP}{FP+TN} \quad (39)$$

4.1.7 FNR

The fraction of true positive samples that were mistakenly labelled as negative is represented by FNR. It assesses the likelihood of the model to classify positive samples as negative.

$$FNR = \frac{FN}{TP+FN} \quad (40)$$

4.1.8 MCC

The MCC scales from -1 to +1 and combines information about true and false positives and negatives into a single value. Whereas +1 signifies a perfect classification, 0 denotes random categorization, and -1 denotes the complete disagreement between observation and prediction, the MCC ranges from -1 to +1.

$$MCC = \frac{((TP*TN)-(FP*FN))}{\sqrt{((TP+FP)*(TP+FN)*(TN+FP)*(TN+FN)}} \quad (41)$$

4.1.9 NPV

*I*NPV determines the proportion of actual negative samples that were correctly classified as such out of all samples that were anticipated to be negative.

$$NPV = \frac{TN}{TN+FN} \quad (42)$$

where the number of correctly classified positively classified samples is represented by TP (True Positives), the number of correctly classified negatively classified samples is represented by TN (True Negatives), and the number of incorrectly classified negatively classified samples is represented by FP (False Positives), FN (False Negatives), and so on.

Table 1 shows Comparison between the suggested and current methods.

Table 1: Comparison between the suggested and current methods

Metrics	Proposed	ANN	CNN	FFNN
Sensitivity	0.978302	0.958449	0.966321	0.947053
Specificity	0.905238	0.809507	0.889564	0.823241
Accuracy	0.961715	0.924636	0.948896	0.918945
Precision	0.972339	0.944848	0.967525	0.948034
Recall	0.978302	0.958449	0.966321	0.947053
F-measures	0.975312	0.9516	0.966923	0.947543
NPV	0.924545	0.851229	0.88581	0.820351
FPR	0.094762	0.190493	0.110436	0.176759
FNR	0.021698	0.041551	0.033679	0.052947
MCC	0.890187	0.78189	0.854609	0.769339

The proposed deep learning model is compared with ANN, CNN, and FFNN, three current models based on the CIC-DDoS2019 dataset. The following are some of the important metrics to be employed in this situation: F-measure, NPV, FPR, FNR, Sensitivity, Specificity, Accuracy, Precision, Recall, and MCC. The proposed model performs better than CNN (0.966321), ANN (0.958449), and FFNN (0.947053) with the maximum sensitivity of 0.978302. With 0.905238, this has the highest specificity while exhibiting a higher propensity for false alarms. At 96.17%, the model's accuracy surpasses that of ANN, CNN, and FFNN, which are at 92.46%, 94.89%, and 91.89%, respectively. The model with the highest precision, with a value of 0.972339, also has the fewest false positives in comparison to other models. With an F-measure score of 0.975312, which is closely followed by CNN at 0.966923, the model with the lowest NPR is still the best one. This model also shows the highest NPV, which is 0.924545; FFNN showed the lowest NPV. The smallest FPR and FPR, which are considered to be negligible mistakes, are 0.094762 and 0.021698, correspondingly. The model's MCC value of 0.890187 indicates that the actual and projected classifications are most closely connected. When compared to ANN, CNN, and FFNN, the suggested model outperforms them all three metrics by a wide margin.

4.2 Graphical representation of performance metrics

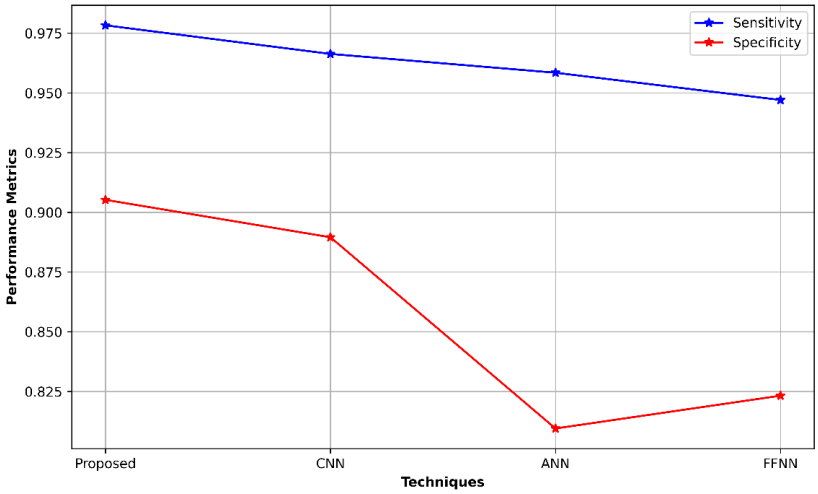

Figure 4: Analysis of sensitivity and specificity

Figure 4 present the performance metric is Specificity and sensitivity, which shows how well the model simulates real-world DDoS attacks; higher values correspond to more accurate detection. By achieving a high value of

0.978302, this model outperforms CNN, ANN, and FFNN in terms of sensitivity. Specificity reveals the model's capability to precisely categorize typical traffic. In the same dimension, the suggested model fared better than the others, achieving a specificity score of 0.905238, suggesting that it may have fewer false positives and be more accurate in real categorized traffic.

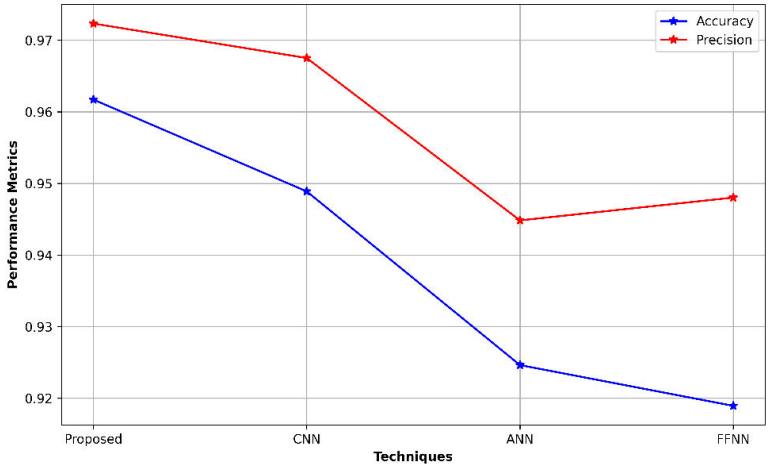

Figure 5: Analysis of Accuracy and Precision

Accuracy is a key indicator of a model's overall effectiveness, representing the proportion of correctly identified instances, both positive and negative. The proposed model excels with an accuracy of 96.17%, outperforming ANN (92.46%), CNN (94.89%), and FFNN (91.89%), highlighting its superior performance in detecting both DDoS attacks and normal traffic. Precision, which focuses on the accuracy of positive predictions, Figure 5 shows the proposed model achieving the highest value (0.972339), meaning it generates fewer false positives compared to the other models, ensuring more reliable identification of actual threats.

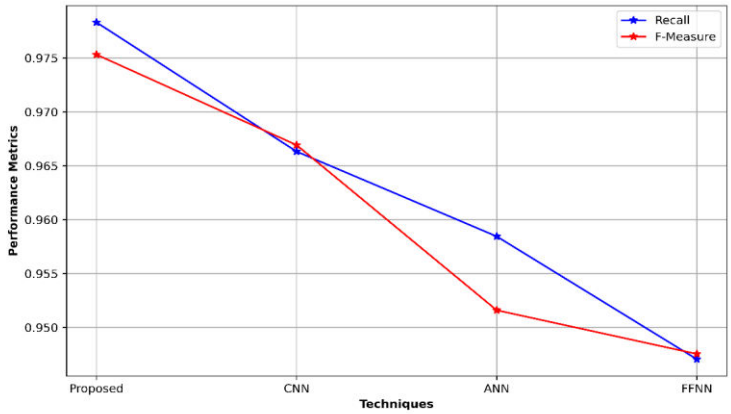

Figure 6: Analysis of Recall and F measures

Figure 6 depicts the graphical analysis of recall and F-measure. The proposed model beats the others in predicting assault scenarios, achieving the highest recalls (value = 0.978302). The values for ANN, FFNN, and CNN are 0.947053, 0.958449, and 0.966321, in that order. The F-measure is utilised to compute the harmonic mean of recall and precision. It provides a neutral assessment of the entire show. With the F-measure attaining a maximum value of 0.975312, the proposed model was demonstrated to be highly proficient and more successful than others in identifying attacks and minimizing false positives.

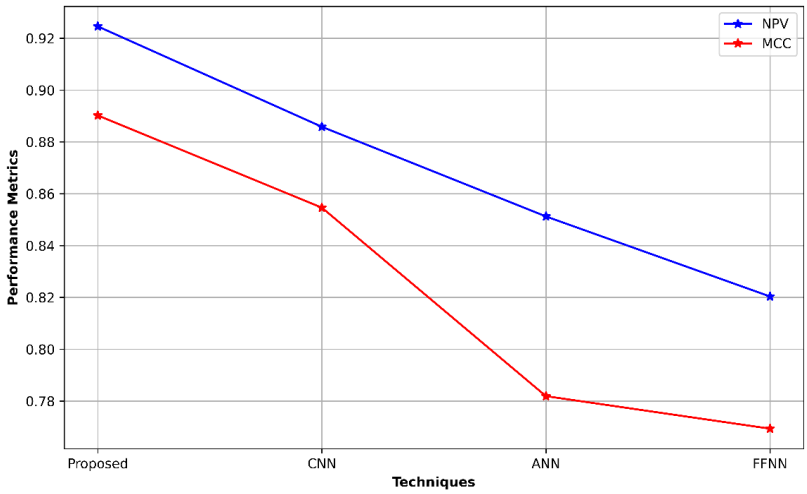

Figure 7: Analysis of NPV and MCC

Figure 7 presented the NPV of 0.924545, the model has good accuracy in detecting typical traffic and preventing false negatives. With a high MCC value of 0.890187, this model exhibits robust and dependable classification performance in contrast to other models such as ANN, CNN, and FFNN, which have large probabilities of false positives. The suggested model is preferable in producing accurate and dependable outcomes because the MCC values of ANN, CNN, and FFNN are lower, suggesting their less efficacy in balancing prediction and classification.

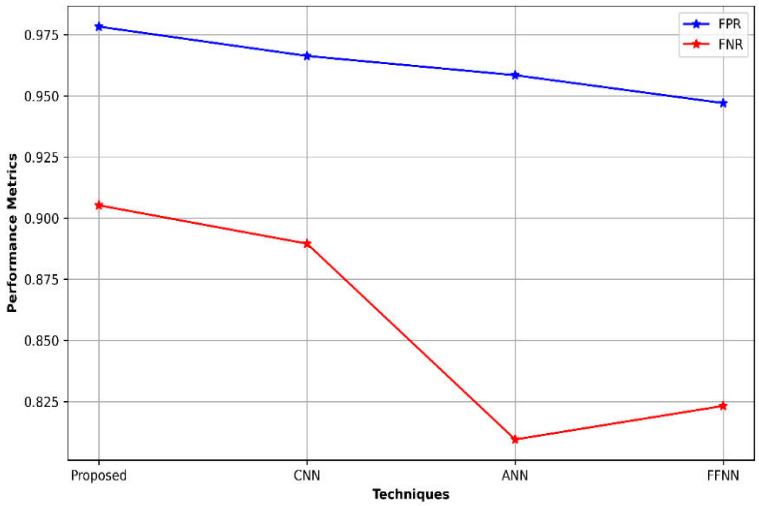

Figure 8: Analysis of FPR and FNR

The Figure 8 proposed model effectively lowers false alarms and distinguishes DDoS assaults from normal traffic with a low FPR of 0.094762. Its performance is notably inferior than other models like ANN, CNN, and FFNN, which are more prone to errors and false alarms. The FNR, which has a low value of 0.021698, also shows how well the model can detect most attacks without overlooking any significant risks. This methodology has the advantage of improving detection overall by minimizing false positives and utilizing the generic security features provided by cloud settings.

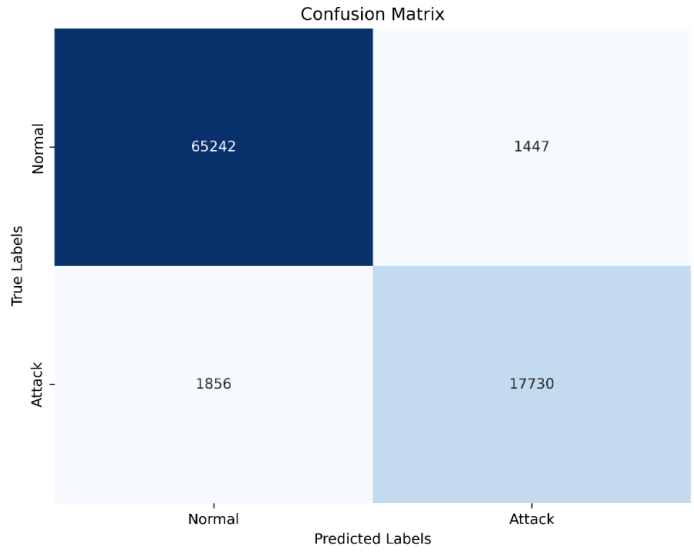

Figure 9: Analysis of Confusion matrix

Figure 9 provide the confusion matrix, the model's classification performance is examined. In cases where the actual label is Attack, the model predicts and accurately identifies instances of Attack. On the other hand, false positives and positives indicate that the model has problems correctly classifying Attack situations. With a normal classification rate of 65,242, the model has a greater accuracy rate than misclassified ones; nonetheless, for the "Attack" example, there is still potential for improvement in false positives and negatives. The model's accuracy and recall can be evaluated using this string performance analysis, especially on "Attack" scenarios. When compared to incorrectly classified cases, the model's accuracy rate for "Normal" classification is greater; nevertheless, for the "Attack" scenario, there is still potential for improvement in terms of false positives and negatives.

5. CONCLUSION

In conclusion, this research aims to improve the detection of DDoS attacks on cloud computing by establishing a novel deep learning framework with optimization capabilities. The current traditional approaches are severely limited by scalability and real-time analysis concerns; however, these concerns do not appear to be related to the GGMNON for classification and the IOOA for feature selection in the context of the proposed system. Thus, the essential statistical traits that were extracted from the input dataset by careful preprocessing give a strong basis for its excellent performance. The experiments showed that the framework outperforms state-of-the-art techniques in terms of accuracy noticeably higher accuracy and efficiency for the detection of anomalies in traffic patterns. This outcome strengthens our model's position as a reliable means of bolstering cloud security against DDoS assaults and enables enterprises to take use of cloud computing's advantages while retaining an even stronger barrier against DDoS cyberattacks. Subsequent research endeavours may involve evaluating the applicability of this paradigm in other cybersecurity contexts and incorporating more machine learning techniques to enhance detection capabilities and system resilience.

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 fund 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.

REFERENCE

[1] Agrawal, N., & Tapaswi, S. (2021). An SDN-assisted defense mechanism for the shrew DDoS attack in a cloud computing environment. *Journal of Network and Systems Management*, *29*(2), 12.

[2] Ahmim, A., Maazouzi, F., Ahmim, M., Namane, S., & Dhaou, I. B. (2023). Distributed denial of service attack detection for the Internet of Things using hybrid deep learning model. *IEEE Access*, *11*, 119862-119875.

[3] Akgun, D., Hizal, S., & Cavusoglu, U. (2022). A new DDoS attacks intrusion detection model based on deep learning for cybersecurity. *Computers & Security*, *118*, 102748.

[4] Aktar, S., & Nur, A. Y. (2023). Towards DDoS attack detection using deep learning approach. *Computers & Security*, *129*, 103251.

[5] Alduailij, M., Khan, Q. W., Tahir, M., Sardaraz, M., Alduailij, M., & Malik, F. (2022). Machine-learning-based DDoS attack detection using mutual information and random forest feature importance method. *Symmetry*, *14*(6), 1095.

[6] Bhardwaj, A., Mangat, V., & Vig, R. (2020). Hyperband tuned deep neural network with well posed stacked sparse autoencoder for detection of DDoS attacks in cloud. *IEEE Access*, *8*, 181916-181929.

[7] Cil, A. E., Yildiz, K., & Buldu, A. (2021). Detection of DDoS attacks with feed forward based deep neural network model. *Expert Systems with Applications*, *169*, 114520.

[8] Doriguzzi-Corin, R., Millar, S., Scott-Hayward, S., Martinez-del-Rincon, J., & Siracusa, D. (2020). LUCID: A practical, lightweight deep learning solution for DDoS attack detection. *IEEE Transactions on Network and Service Management*, *17*(2), 876-889.

[9] Doshi, K., Yilmaz, Y., & Uludag, S. (2021). Timely detection and mitigation of stealthy DDoS attacks via IoT networks. *IEEE Transactions on Dependable and Secure Computing*, *18*(5), 2164-2176.

[10] El Kafhali, S., El Mir, I., & Hanini, M. (2022). Security threats, defense mechanisms, challenges, and future directions in cloud computing. *Archives of Computational Methods in Engineering*, *29*(1), 223-246.

[11] Elsaeidy, A. A., Jamalipour, A., & Munasinghe, K. S. (2021). A hybrid deep learning approach for replay and DDoS attack detection in a smart city. *IEEE Access*, *9*, 154864-154875.

[12] Garcia, J. F. C., & Blandon, G. E. T. (2022). A deep learning-based intrusion detection and preventation system for detecting and preventing denial-of-service attacks. *IEEE Access*, *10*, 83043-83060.

[13] Hasan, T., Malik, J., Bibi, I., Khan, W. U., Al-Wesabi, F. N., Dev, K., & Huang, G. (2022). Securing industrial internet of things against botnet attacks using hybrid deep learning approach. *IEEE Transactions on Network Science and Engineering*, *10*(5), 2952-2963.

[14] Hussain, B., Du, Q., Sun, B., & Han, Z. (2020). Deep learning-based DDoS-attack detection for cyber–physical system over 5G network. *IEEE Transactions on Industrial Informatics*, *17*(2), 860-870.

[15] Islam, M. N. U., Fahmin, A., Hossain, M. S., & Atiquzzaman, M. (2021). Denial-of-service attacks on wireless sensor network and defense techniques. *Wireless Personal Communications*, *116*, 1993-2021.

[16] Najafimehr, M., Zarifzadeh, S., & Mostafavi, S. (2022). A hybrid machine learning approach for detecting unprecedented DDoS attacks. *The Journal of Supercomputing*, *78*(6), 8106-8136.

[17] Novaes, M. P., Carvalho, L. F., Lloret, J., & Proença Jr, M. L. (2021). Adversarial Deep Learning approach detection and defense against DDoS attacks in SDN environments. *Future Generation Computer Systems*, *125*, 156-167.

[18] Perez-Diaz, J. A., Valdovinos, I. A., Choo, K. K. R., & Zhu, D. (2020). A flexible SDN-based architecture for identifying and mitigating low-rate DDoS attacks using machine learning. *IEEE Access*, *8*, 155859-155872.

[19] Priyadarshini, R., & Barik, R. K. (2022). A deep learning based intelligent framework to mitigate DDoS attack in fog environment. *Journal of King Saud University-Computer and Information Sciences*, *34*(3), 825-831.

[20] Ray, S., Mishra, K. N., & Dutta, S. (2023). An Innovative Technique for DDoS Attack Recognition and Deterrence on M-Health Sensitive Data. *Wireless Personal Communications*, *128*(3), 1763-1797.

[21] SaiSindhuTheja, R., & Shyam, G. K. (2021). An efficient metaheuristic algorithm based feature selection and recurrent neural network for DoS attack detection in cloud computing environment. *Applied Soft Computing*, *100*, 106997.

[22] Virupakshar, K. B., Asundi, M., Channal, K., Shettar, P., Patil, S., & Narayan, D. G. (2020). Distributed denial of service (DDoS) attacks detection system for OpenStack-based private cloud. *Procedia Computer Science*, *167*, 2297-2307.

[23] Wei, Y., Jang-Jaccard, J., Sabrina, F., Singh, A., Xu, W., & Camtepe, S. (2021). Ae-mlp: A hybrid deep learning approach for ddos detection and classification. *IEEE Access*, *9*, 146810-146821.

[24] Xu, Y., Deng, G., Zhang, T., Qiu, H., & Bao, Y. (2021). Novel denial-of-service attacks against cloud-based multi-robot systems. *Information Sciences*, *576*, 329-344.

[25] Yungaicela-Naula, N. M., Vargas-Rosales, C., & Perez-Diaz, J. A. (2021). SDN-based architecture for transport and application layer DDoS attack detection by using machine and deep learning. *IEEE Access*, *9*, 108495-108512.