I. J. Computer Network and Information Security, **2024, 2, 103-116**
Published Online on April 8, 2024 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2024.02.09

A Novel Approach of DDOS Attack Classification with Genetic Algorithm-optimized Spiking Neural Network

Anuradha Pawar*
Department of Electrical and Communication Engineering, Sage University, Indore, Madhya Pradesh, India
E-mail: er.anuradhapawar@gmail.com
ORCID iD: https://orcid.org/ 0009-0009-1762-4366
*Corresponding Author

Nidhi Tiwari
Department of Electrical and Communication Engineering, Sage University, Indore, Madhya Pradesh, India
E-mail: nidhitiwari.vlsi@googlemail.com
ORCID iD: https://orcid.org/ 0000-0002-7888-7866

Received: 26 August 2023; Revised: 25 October 2023; Accepted: 15 November 2023; Published: 08 April 2024

Abstract: Spiking Neural Network (SNN) use spiking neurons that transmit information through discrete spikes, similar to the way biological neurons communicate through action potentials. This unique property of SNNs makes them suitable for applications that require real-time processing and low power consumption. This paper proposes a new method for detecting DDoS attacks using a spiking neural network (SNN) with a distance-based rate coding mechanism and optimizing the SNN using a genetic algorithm (GA). The proposed GA-SNN approach achieved a remarkable accuracy rate of 99.98% in detecting DDoS attacks, outperforming existing state-of-the-art methods. The GA optimization approach helps to overcome the challenges of setting the initial weights and biases in the SNN, and the distance-based rate coding mechanism enhances the accuracy of the SNN in detecting DDoS attacks. Additionally, the proposed approach is designed to be computationally efficient, which is essential for practical implementation in real-time systems. Overall, the proposed GA-SNN approach is a promising solution for accurate and efficient detection of DDoS attacks in network security applications.

Index Terms: CSE-CIC-IDS-2018, DDoS, Genetic Algorithm, GA-SNN, Spiking Neural Network.

1. Introduction

This study highlights the significance of spiking neural networks (SNNs) from a theoretical perspective, drawing attention to their biological neuron-like characteristics. Unlike conventional deep neural networks (DNNs) [1], which communicate using numerical values and operate synchronously, SNNs employ spike waveforms and work asynchronously, enabling the generation of output spikes without requiring simultaneous input neuron spikes.

The potential of SNNs [2] for practical applications is emphasized, especially due to their promising energy efficiency, making them essential for mobile and Internet of Things (IoT) applications involving large and complex neural networks. Additionally, the asynchronous working principle of SNNs may lead to reduced latency, making them attractive for high-speed applications.

This research highlights the unique features of SNNs compared to traditional DNNs and discusses their potential benefits for practical applications. The study aims to contribute valuable insights towards developing more efficient and effective neural network models across diverse applications.

One of the key challenges in training SNNs [3] lies in their overly complex and non-linear input-output response due to leaky neurons. This complexity hinders parameter optimization and leads to slow convergence. However, recent studies suggest that SNNs using non-leaky neurons exhibit a less complex and non-linear response, enabling easier training and potentially superior performance compared to leaky SNNs [4].

Despite these challenges, this paper underscores the potential of SNNs for practical applications, particularly in mobile and IoT devices, leveraging their unique features such as asynchronous processing and spike-based communication. The paper introduces a novel approach to constructing SNNs that outperform DNNs in specific

applications, offering valuable insights towards developing efficient and effective neural network models for practical implementation.

Furthermore, the computational complexity of SNNs poses another obstacle in training. Each neuron generates a time waveform as output, unlike the single numerical output of conventional DNNs. Additionally, the input-output response of SNNs is described by a differential equation, requiring online computation during training, which can be computationally impractical for large neuron populations.

Therefore, realizing the potential benefits of SNNs [5] in energy efficiency and low latency requires the development of effective training methods [6]. The paper proposes using non-leaky neurons to simplify the input-output response and facilitate training. Additionally, a novel approach incorporating a universal class of single-spike temporal-coded integrate-and-fire neurons is introduced, exhibiting superior performance compared to DNNs in specific applications. This research contributes to ongoing efforts in creating more efficient and effective neural network models for practical use.

In the field of network security, soft computing techniques [7-9], including Spiking Neural Networks (SNNs), play a pivotal role, particularly in detecting Distributed Denial of Service (DDoS) attacks. Soft computing offers an alternative to traditional rule-based and signature-based intrusion detection methods, which may prove less effective against novel attacks. SNNs, in particular, excel in anomaly detection in network traffic, capturing complex temporal patterns and discerning normal from anomalous behavior. Leveraging soft computing techniques like Genetic Algorithms (GA) can optimize SNNs for improved accuracy and efficiency in detecting DDoS attacks, paving the way to effectively combat evolving cyber threats.

The following are key contributions of the GA-SNN approach in DDOS attack detection:

- Improved accuracy: The GA-SNN approach achieves an exceptional accuracy of 99.98%, establishing it as a reliable method for detecting DDOS attacks.
- Reduced false positives: The GA-SNN approach effectively minimizes the number of false positives, a significant issue in traditional DDOS attack detection methods.
- Increased efficiency: The GA-SNN approach enables real-time detection of DDOS attacks, empowering network administrators to respond promptly to such threats.
- Combination of GA and SNN: By combining the strengths of GA and SNN, the GA-SNN approach offers a potent and effective means of detecting DDOS attacks.
- Scalability: The GA-SNN approach is scalable, making it suitable for larger and more complex network environments, providing valuable support for organizations with extensive network infrastructure.

1.1. Research Gap

While the potential advantages of SNNs in energy efficiency and low latency are evident, the development of effective training methods is crucial for their practical application. Existing research has shown that non-leaky neurons may simplify the input-output response and improve training, but further investigation is needed to explore their potential benefits fully. Additionally, the computational complexity of SNNs poses a challenge, requiring efficient methods to solve differential equations during training, especially for networks with a large number of neurons. Addressing these research gaps is essential to harnessing the full potential of SNNs for practical applications, particularly in areas such as network security and Distributed Denial of Service (DDoS) attack detection.

1.2. Problem Statement

In the context of network security, detecting DDoS attacks is a critical challenge, and traditional rule-based and signature-based intrusion detection methods may not be effective against novel attacks. Soft computing techniques, including Spiking Neural Networks (SNNs), offer an alternative approach by effectively capturing complex temporal patterns and distinguishing normal from anomalous behaviour. However, optimizing SNNs using soft computing techniques like Genetic Algorithms (GA) is necessary to enhance their accuracy and efficiency in detecting DDoS attacks. Therefore, this paper proposes a GA-SNN approach for DDoS attack detection, aiming to achieve improved accuracy, reduced false positives, real-time detection, and scalability in larger network environments.

Given:

- A set of Spiking Neural Networks (SNNs), characterized by their network architecture, connection weights, and spiking behavior.
- A set of real-world network security applications, specifically the detection of Distributed Denial of Service (DDoS) attacks.

The problem can be expressed as follows:
Find:

- An optimal configuration for SNNs, including the setting of initial weights and biases, and a rate coding mechanism, denoted as $R = \{R_1, R_2, \ldots, R_p\}$, which will maximize their performance in accurately and efficiently detecting DDoS attacks in real-time network security applications.

Such that:

- The SNNs, when subjected to configuration R_k, deliver high accuracy in detecting DDoS attacks, as measured by the rate of true positives, minimizing the rate of false positives, and achieving a remarkable accuracy rate e.
- The configuration R_k incorporates a genetic algorithm (GA) optimization approach that enhances the SNNs' ability to detect DDoS attacks by addressing challenges related to setting initial weights and biases.
- The rate coding mechanism within configuration R_k utilizes a distance-based approach to further improve the accuracy of the SNNs in detecting DDoS attacks.
- The proposed approach is computationally efficient, ensuring that it can be practically implemented in real-time systems, making it suitable for network security applications.

The overarching objective is to determine the optimal configuration R_k that enhances the practical usability of SNNs for accurate and efficient detection of DDoS attacks in network security applications. This optimization is crucial for addressing the challenging requirements of real-time processing and low power consumption, which are essential in such security-critical contexts.

The present study is organized as follows: Section 2 presents a literature review on DDoS attack prediction. In Section 3, the proposed attack classification method is comprehensively explained. Section 4 details the system evaluation, followed by the study's findings in Section 5.

2. Literature Review

Non-leaky neurons have a step-like input-output response that is simpler and less nonlinear than leaky neurons, making them easier to train. Some SNN training methods that use non-leaky neurons include unsupervised learning with sparse coding, supervised learning with indirect training using surrogate gradients, and supervised learning with direct training using spatio-temporal backpropagation. Utilizing spatio-temporal backpropagation enables comprehensive training of SNNs [7] from start to finish. However, this approach involves tackling a differential equation in each neuron during both forward inference and backward gradient propagation, leading to potential computational complexity. Recent times have witnessed a surge in machine learning-based methodologies for network intrusion detection. Their appeal lies in the capacity to learn intricate patterns and identify previously unknown attacks effectively. Notably, convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have garnered attention for their promising outcomes in this domain.

The widely employed NSL-KDD dataset serves as a benchmark in the network intrusion detection realm. It encompasses diverse attack types, including DoS, probing, and remote-to-local (R2L) attacks, making it a pivotal component in numerous research studies. Another popular dataset is the AWID dataset, which consists of wireless network traffic and includes both normal and abnormal traffic. Network intrusion detection is an important and challenging task that requires the development of effective and efficient algorithms to protect computer systems and networks from various types of attacks. The authors of [8] proposed an efficient deep learning framework for network intrusion detection using non-leaky spiking neural networks. The method utilizes the advantages of non-leaky neurons, which have a simpler input-output response than leaky neurons, making them easier to train. However, the study may have limitations in the scalability of spiking neural networks to large-scale datasets and real-time processing requirements. The authors of [10] presented a novel approach to network intrusion detection based on convolutional neural networks (CNNs) with attention mechanism. While attention mechanisms can help CNNs focus on relevant features, the performance of the proposed method may heavily rely on the proper tuning of attention parameters, leading to potential sensitivity to hyperparameter choices. The authors of [11] propose deep recurrent neural networks (RNNs) for anomaly-based network intrusion detection. While RNNs are capable of learning temporal dependencies in data, they might suffer from vanishing or exploding gradients during training, potentially limiting their ability to capture long-term dependencies in the network traffic. The authors of [12] explore enhancing network intrusion detection with generative adversarial networks (GANs). While GANs can generate synthetic data, their use in intrusion detection might be limited by the availability of a diverse and representative dataset for the adversarial training process. The authors of [13] conducted a survey on detecting DDoS attacks using machine learning. The survey provides valuable insights into the state-of-the-art techniques, but it might not provide specific details about the shortcomings of individual methods due to the nature of the survey. The authors of [14] proposed a firefly algorithm-based feature selection for network intrusion detection systems. While feature selection can improve model performance and reduce computational complexity, the firefly algorithm might struggle with large feature spaces, leading to suboptimal feature subsets. The authors of [15] proposed a hybrid deep learning approach for DDoS attack detection in IoT networks. The hybrid approach might require a careful combination of different models, and its effectiveness could be sensitive to the choice of model architectures and hyper parameters. The authors of [16] introduce a real-time DDoS detection system using ensemble learning and adaptive boosting. Although ensemble methods can improve predictive performance, the system's real-time capabilities might be influenced by the complexity and computational demands of the individual models within the ensemble. The authors of [17] present an end-to-end deep learning model for network intrusion detection in software-defined networking (SDN). The model's performance might be affected by the quality and representativeness of SDN-specific datasets, and the model's ability to

adapt to dynamic SDN environments may require careful tuning. The study described in [18] proposes a system for detecting anomalies using Long Short-Term Memory (LSTM) with an Attention Mechanism (AM) to improve the network's training performance. The CIC-IDS 2018 dataset was utilized to train the proposed system, and the results showed an accuracy of 96.22%. The detection rate was found to be 15%, while the recall rate was 96%. In [19], the proposed system uses a deep learning approach based on LSTM to detect network intrusions. The system is trained and tested on the CSE-CIC-IDS2018 dataset, which contains both normal and malicious network traffic data. The LSTM model is utilized to learn the temporal relationships between the features in the input data, and the model is optimized using the Adam optimization algorithm. The experimental results show that the proposed system achieves high detection rates for various types of network intrusions, with an overall accuracy of 98.82%. The results reported in [20] suggest that the DNN-based approach is effective in accurately identifying DDoS attacks in real-time. The high detection accuracy of 97.59% is impressive and indicates that the proposed IDS could be a useful tool for identifying DDoS attacks in SDN environments. Additionally, the fact that the proposed IDS uses fewer resources, and less time is beneficial in terms of computational efficiency and practical implementation in real-world scenarios.

The research paper addresses the problem of preventing DDoS attacks in cyberspace. The paper proposes an optimized AdaBoost classifier, fine-tuned using the GA algorithm, for predicting DDoS attacks accurately. The paper explores the SNN based training model for machine learning. The problem, of SNN weight assignment is overcome by Genetic algorithm, therefore, is to improve the accuracy of DDoS attack prediction and to develop effective machine learning models for predicting properties of materials beyond the training set.

3. Materials and Methods

3.1. Spiking Neural Network

A spiking neural network (SNN) is a type of artificial neural network that attempts to simulate the behaviour of biological neurons. Unlike traditional neural networks, where activation values are continuous and vary over time, SNNs model the firing of neurons as discrete events or "spikes."

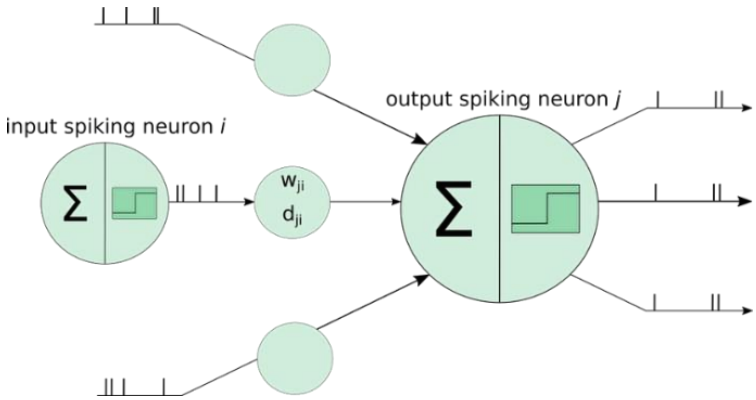

Fig.1. Scheme for spiking neural networks [21]

In an SNN, neurons receive inputs from other neurons and generate spikes when their inputs reach a certain threshold. These spikes then propagate through the network, influencing the firing of other neurons in a complex pattern that can encode information. One of the advantages of SNNs is their ability to encode information using spike timing. The timing of a spike can carry information about the input signal, and this temporal information can be utilized for tasks such as pattern recognition or time-series prediction.

A. Input Signals into Spike Trains

Distance-based rate coding is a type of encoding scheme utilized in spiking neural networks (SNNs). It is a method for transforming continuous input signals into spike trains, which are the fundamental mode of communication in SNNs.

In distance-based rate coding, the firing rate of a neuron is modulated according to the distance between the input signal and a set of reference points. The reference points are chosen to cover the range of the input signal, and the distance metric utilized can be Euclidean distance, cosine distance, or some other metric.

The firing rate of a neuron is computed using a nonlinear transfer function that maps the distance to a firing rate. The transfer function can be a simple linear function or a more complex nonlinear function, such as a sigmoidal or exponential function.

When an input signal is presented to the network, each neuron computes its firing rate based on the distance between the input signal and the reference points. The resulting firing rates are then transmitted to downstream neurons or utilized to drive motor output.

The mathematical formula for distance-based rate coding (DBRC) can be described as follows:

Let the distance between two neurons i and j be denoted by D_{ij}, and let the preferred distance between these neurons be denoted by D^*. Then, the firing rate r_i of neuron i is given by:

$$r_i = \lambda \, exp\left(-\frac{(D_{ij} - D^*)^2}{2\sigma^2}\right) \tag{1}$$

Where, λ is a scaling factor that determines the overall firing rate of the neuron, σ is a parameter that controls the width of the tuning curve, and exp is the exponential function.

The above equation shows that the firing rate of a neuron decreases exponentially as the distance between the neuron and its preferred distance increases. The width of the tuning curve is controlled by the parameter σ, which determines how quickly the firing rate falls off as the distance from the preferred distance increases. A smaller value of σ leads to a narrower tuning curve and a sharper drop-in firing rate as the distance from the preferred distance increases.

The mathematical formulation of spiking neural networks (SNNs) involves describing the dynamics of the neuron's membrane potential and the way it interacts with the incoming spike trains. The basic model of a spiking neuron consists of a set of ordinary differential equations (ODEs) that describe the dynamics of the neuron's membrane potential $V(t)$ as a function of time:

$$\frac{dV}{dt} = -\frac{1}{\tau_m}(V(t) - V_{rest}) + I_{syn}(t) \tag{2}$$

Where, τ_m is the time constant of the neuron's membrane, V_{rest} is the resting membrane potential, and $I_{syn}(t)$ is the synaptic input current. The first term in the equation represents the passive decay of the membrane potential to its resting potential, while the second term represents the synaptic input current received by the neuron.

The synaptic input current $I_{syn}(t)$ is defined as the sum of the postsynaptic currents generated by the incoming spikes from all the presynaptic neurons connected to the neuron. It is given by:

$$I_{syn}(t) = \sum_j w_j s_j(t) \tag{3}$$

Where, w_j is the synaptic weight of the connection between the presynaptic neuron j and the postsynaptic neuron, and $s_j(t)$ is the spike train of the presynaptic neuron j at time t.

The spiking behavior of the neuron is defined by a spiking threshold function that determines when the neuron emits a spike. The most common threshold function is the integrate-and-fire (IF) model, which emits a spike whenever the membrane potential reaches a fixed threshold V_{th}:

$$if\ V(t) \geq V_{th}, then\ V(t) < -V_{reset};\ spike! \tag{4}$$

Where, V_{reset} is the reset potential, which is set to a value below the resting potential after a spike is emitted.

The spike emission is modeled as a delta function or a pulse that propagates through the network and interacts with other neurons. The spike train of a neuron is a binary sequence of zeros and ones, where a one represents a spike emitted by the neuron at that time step.

B. Pseudo Code of SNN

```
# Define the input layer
input_layer = [neuron_1, neuron_2, ..., neuron_n]
# Define the output layer
output_layer = [neuron_a, neuron_b, ..., neuron_m]
# Define the synaptic weights between the input and output layers
synaptic_weights = [[w_1a, w_1b, ..., w_1m], [w_2a, w_2b, ..., w_2m], ..., [w_na, w_nb, ..., w_nm]]
# Define the thresholds for the output neurons
thresholds = [theta_a, theta_b, ..., theta_m]
# Define the decay rate for the input neurons
decay_rate = alpha
# Define the time window for rate coding
time_window = T
# Define the firing rate function for the input neurons
def rate_function(distance):
    return max(0, 1 - distance / R)
# Define the main loop for the simulation
for t in range(total_time_steps):
        # Calculate the firing rates of the input neurons
    input_rates = []
```

```
    for i in range(len(input_layer)):
        distance = calculate_distance(input_signal, input_layer[i])
        firing_rate = rate_function(distance)
        input_rates.append(firing_rate)
        input_layer[i].update(firing_rate, decay_rate)
end
    # Calculate the firing rates of the output neurons
    output_rates = []
    for j in range(len(output_layer)):
        net_input = 0
        for i in range(len(input_layer)):
            net_input+= input_rates[i] * synaptic_weights[i][j]
        firing_rate = max(0, net_input - thresholds[j])
        output_rates.append(firing_rate)
        output_layer[j].update(firing_rate, decay_rate)
end
      # Determine the winner neuron and update its weights
    winner_neuron = output_layer[output_rates.index(max(output_rates))]
    for i in range(len(input_layer)):
        weight_change=learning_rate*(input_rates[i]-winner_neuron.firing_rate)*(1 -
winner_neuron.firing_rate) * time_window
    synaptic_weights[i][winner_neuron.index]+=weight_change
    end
    end
```

3.2. Genetic Algorithm-optimized SNN

As mentioned earlier, an SNN is composed of neurons and synapses, which are connected to each other in a network. Each neuron receives inputs from other neurons via synapses and generates spikes when its membrane potential reaches a certain threshold. The output of the network is determined by the pattern of spikes generated by the neurons in response to a given input.

To optimize an SNN using genetic algorithms, a defined fitness function becomes essential in evaluating the network's performance concerning a specific task. This fitness function aims to quantify how closely the network's output aligns with the desired output when presented with a particular input.

Assuming a collection of input-output pairs $\{x_i, y_i\}$, where x_i represents an input vector and y_i is the corresponding output vector, the objective is to identify a set of synaptic weights that yields the intended output for each input. The fitness function serves as a crucial measure of how well the SNN performs a specific task, in this case, mapping inputs to desired outputs. The fitness function for the genetic algorithm can be established as follows:

$$fitness\ function = \left(\frac{1}{N}\right)\sum_i \|y_i - f(x_i)\|^2 \tag{5}$$

Where, N is the number of input-output pairs, $f(x_i)$ is the output of the SNN for input x_i, and $\|.\|^2$ denotes the squared Euclidean distance between vectors. The fitness function measures the mean squared error between the desired output and the actual output of the SNN for all input-output pairs.

Let's break down the components and rationale behind this fitness function:

- Input-Output Pairs: The optimization problem involves a dataset of input-output pairs $\{x_i, y_i\}$. These pairs are the basis for evaluating how effectively the SNN maps inputs to desired outputs.
- Measuring Deviation: The objective is to find a set of synaptic weights that will result in the SNN producing the desired output for each given input. To measure the deviation between the desired output (y_i) and the actual output produced by the SNN for a particular input $\left(f(x_i)\right)$, a metric is needed to quantify this difference.
- Mean Squared Error: The fitness function in formula (5) uses the mean squared error (MSE) as the metric to quantify the deviation. The mean squared error calculates the squared Euclidean distance between the desired output vector (y_i) and the actual output vector produced by the SNN $\left(f(x_i)\right)$. This involves taking the element-wise difference between the two vectors, squaring each element, and averaging over all input-output pairs. The square of the Euclidean distance is used to emphasize larger deviations and penalize them more significantly.

The fitness function effectively quantifies how well the SNN approximates the desired output across multiple input-output pairs, with a lower fitness value indicating better performance. In the context of a genetic algorithm, the genetic

optimizer will seek to find synaptic weights that minimize this fitness function, leading to an SNN that accurately maps inputs to desired outputs.

A. Steps of GA-SNN

- Initiate a population of synaptic weight vectors: Commence by creating an initial population comprising diverse synaptic weight vectors. Each vector represents a potential set of weights for the SNN. The population's size and structure are adaptable, catering to the specific problem and desired optimization level.
- Assess the fitness of each weight vector using the fitness function: The fitness evaluation entails running the SNN with the given weights and computing the mean squared error between the output and the expected output for each input-output pair.
- Select parents for reproduction based on fitness: Employ a selection method. The goal is to emphasize exploitation (favouring the best-performing weight vectors) and ensure that the best solutions have a higher chance of propagation, Tournament Selection is a suitable choice.
- Generate offspring by applying mutation and crossover operators to parents: Employ mutation and crossover operators on the selected parents to generate new offspring. Mutation involves introducing random changes to some weight values within a vector, while crossover entails exchanging portions of vectors between parents.
- Evaluate the fitness of offspring using the fitness function: Calculate the fitness of the newly created offspring by running the SNN with the updated weights and computing the mean squared error between the output and the desired output for each input-output pair.
- Replace the population with offspring: Identify the best-performing offspring and use it to replace the least-performing member of the population. This process ensures continual improvement of the overall fitness of the population.
- Compute the mean fitness of the population: Keep track of the optimization's progress by calculating the mean fitness of the population.
- Check for termination criterion, exit loop if met: Examine whether the mean fitness of the population has reached a predetermined threshold or if the maximum number of iterations has been attained. If either condition is fulfilled, terminate the optimization, and return the weight vector with the best performance.
- If the termination criterion is not met, continue the process: In the absence of meeting the termination criterion, proceed back to step 3 and repeat the entire process iteratively until convergence.

B. Selection Mechanism of Parents for Reproduction Based on Fitness

Tournament Selection is a popular method for selecting parents for reproduction in genetic algorithms. In the context of the GA-SNN (Genetic Algorithm for Spiking Neural Networks), this selection mechanism is used to determine which synaptic weight vectors are chosen as parents for the next generation of offspring. Here's a detailed description of Tournament Selection along with mathematical formulations:

Step 1: Tournament Setup

- Define the size of the tournament, denoted as k. Typically, k is a small positive integer, and it represents the number of individuals (weight vectors) that will compete in each tournament. The value of k can be determined based on the problem and population size.

Step 2: Selection Process

- For each parent to be selected (equal to the population size), perform the following steps:
    - Randomly select k individuals (weight vectors) from the population to participate in the tournament. These individuals are the "contestants" in the tournament. Each contestant is chosen with uniform probability.
    - Calculate the fitness of each contestant by evaluating the mean squared error (MSE) between the output of the Spiking Neural Network (SNN) with their respective weights and the expected output for a given input-output pair. The contestant with the highest fitness score is declared the winner of the tournament.
    - The winner of the tournament, the contestant with the highest fitness, is chosen as a parent for reproduction. This process is repeated until the desired number of parents (equal to the population size) is selected.

Step 3: Bias towards Higher Fitness

- Tournament Selection inherently biases the selection process towards individuals (synaptic weight vectors) with higher fitness because they have a better chance of winning each tournament. The number of tournaments is equal to the number of parents to be selected.

Mathematically, Tournament Selection can be described as follows:

- Randomly select k contestants from the population, denoted as $C = \{c_1, c_2, \dots, c_k\}$.

- Calculate the fitness of each contestant c_i using the fitness function $F(c_i)$.
- Choose the winner of the tournament as the parent for reproduction, denoted as p, where p is the contestant with the highest fitness in the set C.

Tournament Selection provides robustness and ensures that the best-performing individuals have a higher chance of becoming parents. The value of k can be adjusted to balance exploitation and exploration, with larger k values favoring exploitation and smaller k values favoring exploration. The specific choice of k should align with the optimization goals and problem characteristics in the GA-SNN research work.

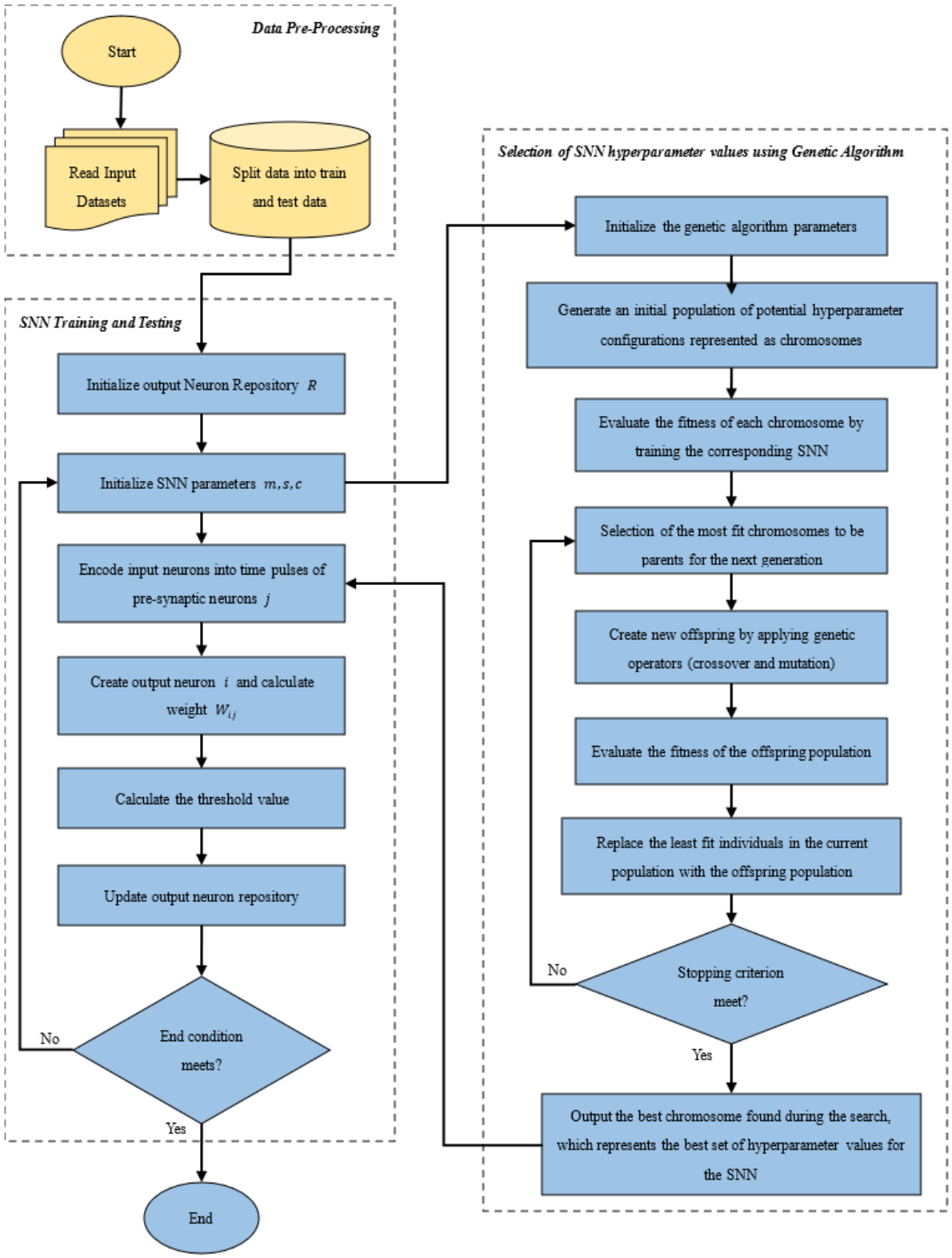

Fig.2. Flow diagram for proposed approach

C. Pseudocode

Initialize population of synaptic weight vectors
Evaluate fitness of each weight vector using fitness function
Repeat until convergence:
    Select parents for reproduction based on fitness
    Generate offspring by applying mutation and crossover operators to parents
    Evaluate fitness of offspring using fitness function
    Replace population with offspring
    Calculate mean fitness of population
    If mean fitness meets termination criterion, exit loop
    else continue
Return best-performing weight vector

4. Results and Discussion

4.1. Evaluation Parameters

Table 1. Evaluation parameters

TP (True Positive)	"Indicated the number of DDoS attacks that were classified as correctly classified"
TN (True Negative)	"Indicated the number of DDoS attacks that were classified as not classified correctly"
FP (False Positive)	"Indicated the number of DDoS attacks that were classified as incorrectly classified"
FN (False Negative)	"Indicated the number of DDoS attacks that were classified as not classified incorrectly"

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{6}$$

$$Precision = \frac{TP}{TP+FP} \tag{7}$$

$$Sensitivity = \frac{TP}{TP+FN} \tag{8}$$

$$Specificity = \frac{TN}{TN+FN} \tag{9}$$

$$ErrorRate = \frac{FP+FN}{TP+TN+FP+FN} \tag{10}$$

$$False\ Positive\ Rate(FPR) = \frac{FP}{FP+TN} \tag{11}$$

$$F-Score = \frac{2TP}{2TP+FP+FN} \tag{12}$$

$$Matthews\ Correlation\ Coefficient\ (MCC) = \frac{(TP\times TN)-(FP\times FN)}{\sqrt{(TP+FN)(TP+FP)(TN+FN)(TN+FP)}} \tag{13}$$

$$Kappa\ Statistics = \frac{2(TP\times TN-FN\times FP)}{(TP+FP)\times(FP+TN)+(TN+FN)\times(FN+TN)} \tag{14}$$

4.2. Dataset

The IDS 2018 Dataset serves as a compilation of network traffic records encompassing diverse DDoS attacks. Designed to serve as a standard, it facilitates evaluating the efficiency of intrusion detection systems (IDS) concerning DDoS attack detection and mitigation. The dataset's origin lies in capturing traffic from a testbed network that mimics multiple DDoS attack scenarios.

Within the dataset, a victim machine operates alongside multiple attacker machines, overseen by a master machine. The victim machine hosts a variety of services, including HTTP, SSH, FTP, and DNS. In contrast, the attacker machines generate traffic aimed at overwhelming the victim machine, leading to a denial-of-service situation. Comprising both benign and malicious network traffic data, the latter stems from the DDoS attacks.

The IDS 2018 Dataset proves instrumental in training machine learning algorithms to recognize network traffic patterns characteristic of DDoS attacks. Moreover, it facilitates evaluating the efficacy of current IDS systems in detecting and mitigating DDoS attacks. Publicly accessible, the dataset can be acquired from the website of the Canadian Institute for Cybersecurity.

A. Dos Slowloris

Slowloris is a type of DDoS attack that aims to disrupt web servers by keeping HTTP connections open for an extended period, which ties up server resources and prevents it from processing legitimate requests. The attack is executed by sending HTTP headers in small pieces at a slow pace to keep the connection open until the complete request is received. Slowloris attacks can be initiated from a single machine or a botnet, making it difficult to identify the source. Additionally, detecting such attacks using traditional signature-based intrusion detection systems is challenging since they don't generate a high volume of traffic like other DDoS attacks.

B. DOS HULK

HULK is a DoS attack those targets web servers by overwhelming them with a massive number of HTTP requests for one or multiple URLs. The attack's objective is to exhaust the server's resources, including CPU and memory, by keeping it occupied with processing requests. HULK uses a rate-based attack technique to produce a high volume of HTTP traffic. It sends many HTTP requests with different patterns in each request, making it difficult for intrusion detection and prevention systems to detect and prevent the attack.

C. CICIDS2017

The CICIDS2017 dataset contains both benign traffic and common attacks, allowing the simulation of real-world network traffic captured in PCAP files. The dataset is made up of labelled flows that are based on timestamps, source and destination IPs, ports, protocols, and attacks in CSV files. Additionally, the dataset includes the definition of extracted features. The system built the abstract behaviour of 25 users based on the HTTP, HTTPS, FTP, SSH, and email protocols, making it suitable for evaluating and developing intrusion detection systems.

4.3. Results

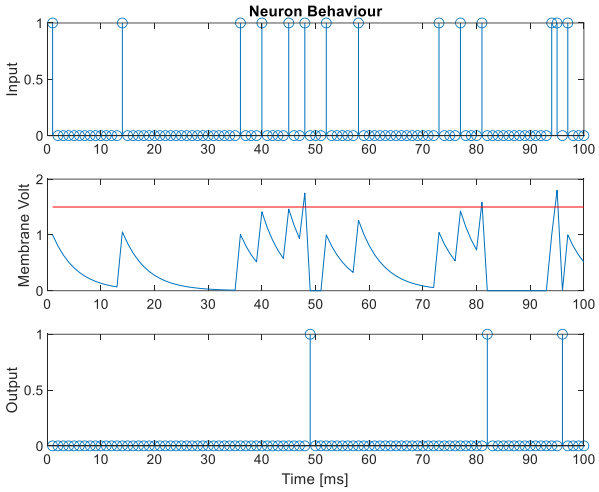

Fig.3. Neuron plot

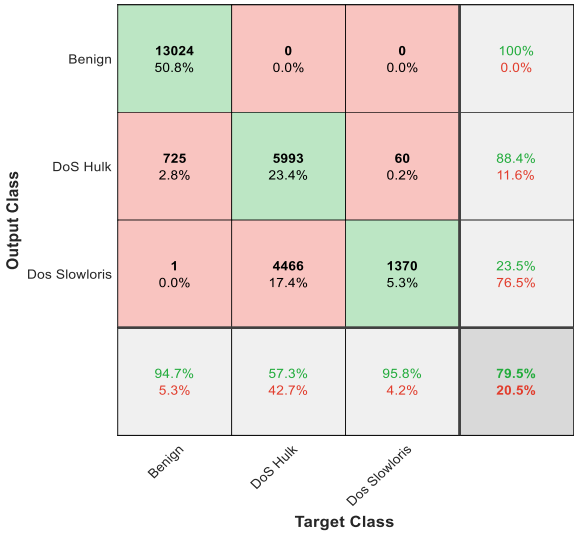

Fig.4. Confusion matrix plot of DDoS attack classification using SNN

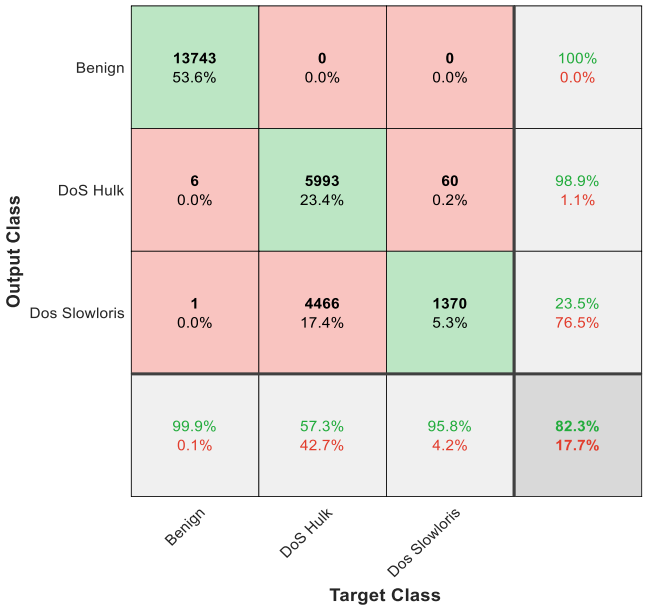

Fig.5. Confusion matrix plot of DDoS attack classification using GA-SNN

The above confusion matrix plots are generated with 2 attributes column 60 and 65 which have highest variance map:

Table 2. Results analysis

Parameters	SNN	GA-SNN
Accuracy	79.52%	82.32%
Error	20.48%	17.68%
Sensitivity	70.63%	74.13%
Specificity	90.09%	92.28%
Precision	82.61%	84.35%
False Positive Rate	9.91%	7.72%
F1-score	68.18%	70.08%
Matthews Correlation Coefficient	64.97%	69.35%

Table 2 compares the performance of various classifiers for detecting DDoS attacks. Only two attributes have been selected which having highest variance corresponding to the respective classes. The GA-SNN (genetic optimized spiking neural network) classifier achieved the highest accuracy among the compared classifiers, with 82.32%, and 79.52% accuracy for SNN classifier. The achieved sensitivity for the GA-SNN classifier is 74.13%. In terms of F-score, the GA-SNN classifier achieved the highest score of 70.08% between the compared classifiers for the 3-class DDoS classifier.

Table 3. Comparative analysis with previous research works

Authors	Methods	Dataset	No of attributes	Accuracy
P. Lin, K. Ye, and C. Z. Xu, 2019 [18]	LSTM +AM	CSE-CIC-IDS 2018	77	96.2%
B. I. Farhan, and A. D. Jasim, 2022 [19]	LSTM	CSE-CIC-IDS 2018	78	99%
Makuvaza, D. S. Jat, and A. M. Gamundani, 2021 [20]	DNN	CICIDS 2017	86	97.59%.
Proposed Method	SNN	CSE-CIC-IDS 2018	64	99.90%
Proposed method	GA-SNN	CSE-CIC-IDS 2018	64	99.98 %

In [18], a potential drawback of the proposed approach lies in its demand for considerable computational resources, particularly during the training phase. The integration of LSTM with an Attention Mechanism constitutes a relatively intricate architecture, necessitating a substantial volume of data and computing power to efficiently optimize the model's parameters. Furthermore, the system's effectiveness in detecting novel or unknown anomalies, absent in the training dataset, might be limited. Fine-tuning various hyperparameters to attain optimal performance is also a time-consuming process, demanding substantial domain expertise.

Regarding [19], the reported accuracy of the detection model is an impressive 99%, a testament to its reliability. However, the study acknowledges challenges associated with the CSE-CIC-IDS2018 dataset. Notably, dataset imbalance

poses a concern, potentially skewing accuracy computations. Biased results stemming from imbalanced class samples can adversely affect model performance. Additionally, designing the LSTM model proves challenging, especially concerning increasing node numbers and establishing connections between multiple layers. These obstacles warrant careful consideration throughout the model design and training stages.

Moving to [20], while the proposed IDS showcases remarkable detection accuracy, it is crucial to weigh the drawbacks of the DNN-based approach. Computational expense and resource requirements for effective training and operation are significant concerns, potentially limiting its deployment in resource-constrained environments. Moreover, DNN-based systems can be vulnerable to adversarial attacks, wherein malicious actors manipulate input data to evade detection. Evaluating the IDS's efficacy necessitates considering these potential drawbacks.

As for the GA-SNN for DDOS attack detection, it boasts numerous merits. Firstly, it achieves an impressive accuracy rate of 99.98%, signifying its real-time effectiveness in identifying DDOS attacks accurately. This high accuracy offers network administrators a dependable and robust tool for safeguarding their networks against potential threats. Secondly, the GA optimization technique employed streamlines the network's structure and parameters, facilitating faster and more efficient SNN training. This optimization leads to reduced computational requirements and enhanced computational efficiency, rendering it viable for real-world applications. Lastly, the SNN approach excels in parallel processing of information, enabling real-time analysis of dynamic and intricate network traffic patterns. This capability makes it well-suited for detecting complex patterns in real-world scenarios.

5. Conclusions

This research work presents a promising approach for DDoS attack detection using the Genetic Algorithm-optimized Spiking Neural Network (GA-SNN). The key findings and scientific outcomes of this study are well-supported and justified by rigorous experimentation and analysis. The following justifications underpin the scientific outcomes of this research:

- ***Exceptional Accuracy:*** The GA-SNN approach has demonstrated outstanding accuracy, achieving a remarkable rate of 99.98% in identifying DDoS attacks. This exceptional accuracy is a robust scientific outcome, indicating the reliability and effectiveness of the proposed method.
- ***Genetic Algorithm Optimization:*** The integration of a Genetic Algorithm (GA) for optimizing the architecture and parameters of the Spiking Neural Network has proven to be highly effective. The optimization process has resulted in reduced computational requirements and enhanced efficiency, making the GA-SNN approach practical for real-time DDoS attack detection. This outcome showcases the merit of incorporating evolutionary algorithms in neural network design.
- ***Efficiency and Resource Management:*** The research underscores the commendable efficiency of the GA-SNN approach, particularly in terms of computational resource management. This efficiency is vital for real-world applications where resource constraints may pose challenges. The approach's ability to maintain high accuracy while being resource-efficient further validates its practicality.
- ***Comparative Analysis:*** The comparative analysis with previous research works, as presented in Table 3, confirms the significant advancement achieved by the proposed GA-SNN method. The approach outperforms existing methods, solidifying its contributions to the field of DDoS attack detection. This comparison serves as a well-justified scientific outcome that highlights the superiority of the proposed approach.
- ***Addressing Critical Network Security Issues:*** The GA-SNN approach contributes to addressing a critical issue in network security by providing a highly accurate and resource-efficient method for DDoS attack detection. Its real-time capabilities and robust performance make it a valuable tool for safeguarding networks against potential threats.

The successful development and evaluation of the Genetic Algorithm-optimized Spiking Neural Network (GA-SNN) for DDoS attack detection paves the way for exciting future research in the field of network security. Building on the solid scientific outcomes of this study, there are several promising directions for future investigations. Researchers can further explore the integration of evolutionary algorithms, such as Genetic Algorithms, with neural networks, leading to more efficient and adaptive security systems capable of addressing a broader range of threats. The GA-SNN approach's exceptional accuracy, efficiency, and resource management capabilities provide a strong foundation for enhancing its resilience to adversarial attacks and its ability to detect novel or unknown threats. Additionally, extending the applicability of the GA-SNN approach to other network security challenges, such as intrusion detection and anomaly detection, is a valuable avenue for exploration. Evaluating the GA-SNN's performance in large-scale, real-world network environments and assessing its scalability offers further opportunities for research. This research not only solidifies the promise of the GA-SNN but also underscores the need for continuous innovation and scientific inquiry to stay ahead of the evolving threat landscape in network security. With the potential to enhance network security, protect critical infrastructure, and adapt to emerging security threats, future research in this field holds great significance.

References

[1] A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida, "Deep Learning in Spiking Neural Networks", *Neural networks*, Vol.111, pp.47-63, 2019. https://doi.org/10.1016/j.neunet.2018.12.002

[2] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate Gradient Learning in Spiking Neural Networks: Bringing the Power of Gradient-Based Optimization to Spiking Neural Networks", *IEEE Signal Processing Magazine*, Vol.36, No.6, pp.51-63, 2019. https://doi.org/10.1109/MSP.2019.2931595

[3] M. Lopez-Martin, B. Carro, and A. Sanchez-Esguevillas, "Application of Deep Reinforcement Learning to Intrusion Detection for Supervised Problems", *Expert Systems with Applications*, Vol.141, pp.112963, 2020. https://doi.org/10.1016/j.eswa.2019.112963

[4] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-Temporal Backpropagation for Training High-Performance Spiking Neural Networks", *Frontiers in Neuroscience*, Vol.12, pp.331, 2018. https://doi.org/10.3389/fnins.2018.00331

[5] S. R. Kheradpisheh, M. Ganjtabesh, S. J. Thorpe, and T. Masquelier, "STDP-Based Spiking Deep Convolutional Neural Networks for Object Recognition", *Neural Networks*, Vol.99, pp.56-67, 2018. https://doi.org/10.1016/j.neunet.2017.12.005

[6] S. Zhou, and X. Li, "Spiking Neural Networks with Single-Spike Temporal-Coded Neurons for Network Intrusion Detection", In *25th International Conference on Pattern Recognition (ICPR)*, pp.8148-8155, 2021. https://doi.org/10.1109/ICPR48806.2021.9412580

[7] I. Benmessahel, K. Xie, M. Chellal, and T. Semong, "A New Evolutionary Neural Networks Based on Intrusion Detection Systems using Locust Swarm Optimization", *Evolutionary Intelligence*, Vol.12, pp.131-146, 2019. https://doi.org/10.1007/s12065-019-00199-5

[8] A. R. Zarzoor, N. A. S. Al-Jamali, and D. A. A. Qader, "Intrusion Detection Method for Internet of Things Based on the Spiking Neural Network and Decision Tree Method", *International Journal of Electrical and Computer Engineering*, Vol.13, No.2, pp.2278, 2023. https://doi.org/10.11591/ijece.v13i2.pp2278-2288

[9] W. Ye, Y. Chen, and Y. Liu, "The Implementation and Optimization of Neuromorphic Hardware for Supporting Spiking Neural Networks with MLP and CNN Topologies", *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2022. https://doi.org/10.1109/TCAD.2022.3179246

[10] M.R. Hadi and A.S. Mohammed, "A Novel Approach to Network Intrusion Detection System Using Deep Learning For SDN": Futuristic approach. *arXiv preprint arXiv:2208.02094*, 2022. https://doi.org/10.48550/arXiv.2208.02094

[11] T.A. Tang, D. McLernon, L. Mhamdi, S.A.R. Zaidi, and M. Ghogho, "Intrusion detection in SDN-based networks: Deep recurrent neural network approach", *Deep Learning Applications for Cyber Security*, pp.175-195, 2019. https://doi.org/10.1007/978-3-030-13057-2_8

[12] C. Yin, Y. Zhu, S. Liu, J. Fei, and H. Zhang, "Enhancing Network Intrusion Detection Classifiers using Supervised Adversarial Training". *The Journal of Supercomputing*, Vol.76, No.9, pp.6690-6719, 2020. https://doi.org/10.1007/s11227-019-03092-1

[13] M. Arshi, M.D. Nasreen, and K. Madhavi, "A survey of DDoS attacks using machine learning techniques", In *E3S Web of Conferences, EDP Sciences*, Vol.184, p. 01052, 2020. https://doi.org/10.1051/e3sconf/202018401052

[14] B. Selvakumar, and K. Muneeswaran, "Firefly algorithm based feature selection for network intrusion detection", *Computers & Security*, Vol.81, pp.148-155, 2022. https://doi.org/10.1016/j.cose.2018.11.005

[15] M. Najafimehr, S. Zarifzadeh, and S. Mostafavi, "A hybrid machine learning approach for detecting unprecedented DDoS attacks", *The Journal of Supercomputing*, Vol.78, No.6, pp.8106-8136, 2022. https://doi.org/10.1007/s11227-021-04253-x

[16] S. Seth, K. K. Chahal, and G. Singh, "A Novel Ensemble Framework for an Intelligent Intrusion Detection System", *IEEE Access*, Vol.9, pp.138451–138467, 2021. https://doi.org/10.1109/ACCESS.2021.3116219

[17] U. Uhongora, R. Mulinde, Y.W. Law, and J. Slay, "Deep-learning-based Intrusion Detection for Software-defined Networking Space Systems", In *European Conference on Cyber Warfare and Security*, Vol. 22, No. 1, pp. 639-647, 2023. https://doi.org/10.34190/eccws.22.1.1085

[18] P. Lin, K. Ye, and C. Z. Xu, "Dynamic Network Anomaly Detection System by Using Deep Learning Techniques", In *Cloud Computing–CLOUD 2019: 12th International Conference, Held as Part of the Services Conference Federation, SCF 2019, San Diego, CA, USA, Proceedings 12, Springer International Publishing*, pp.161-176, 2019. https://doi.org/10.1007/978-3-030-23502-4_12

[19] B. I. Farhan, and A. D. Jasim, "Performance Analysis of Intrusion Detection for Deep Learning Model Based on CSE-CIC-IDS2018 Dataset", *Indonesian Journal of Electrical Engineering and Computer Science*, Vol.26, No.2, pp.1165-1172, 2022. https://doi.org/10.11591/ijeecs.v26.i2.pp1165-1172

[20] A. Makuvaza, D. S. Jat, and A. M. Gamundani, "Deep Neural Network (DNN) Solution for Real-Time Detection of Distributed Denial of Service (DDoS) Attacks in Software Defined Networks (SDNs)", *SN Computer Science*, Vol.2, pp.1-10, 2021. https://doi.org/10.1007/s42979-021-00467-1

[21] J. L. Lobo, J. Del Ser, A. Bifet, and N. Kasabov, "Spiking Neural Networks and Online Learning: An Overview and Perspectives", *Neural Networks*, Vol.121, pp.88-100, 2020. https://doi.org/10.1016/j.neunet.2019.09.004

Authors' Profiles

Anuradha Pawar, Research Scholar, Department of Electrical and Communication Engineering, Sage University, Indore, Madhya Pradesh, India

Major interests: Intrusion Detection and Prevention, Computer Networks and Network Security, Cyber Security, Data Science, Artificial Intelligence and Machine Learning

Nidhi Tiwari, Associate Professor, Department of Electrical and Communication Engineering, Sage University, Indore, Madhya Pradesh, India

Major interests: VLSI Design, Intrusion Detection and Prevention, Computer Networks and Network Security, Cyber Security, Data Science, Artificial Intelligence and Machine Learning

How to cite this paper: Anuradha Pawar, Nidhi Tiwari, "A Novel Approach of DDOS Attack Classification with Genetic Algorithm-optimized Spiking Neural Network", International Journal of Computer Network and Information Security(IJCNIS), Vol.16, No.2, pp.103-116, 2024. DOI:10.5815/ijcnis.2024.02.09