Enhanced Arachnid Swarm-Tuned Convolutional Neural Network Model for Efficient Intrusion Detection

Nishit Patil[1], Dr. Shubhlaxmi Joshi[2]
Research Scholar, School of Computer Science, MIT-WPU, Pune, Maharashtra, India[1]
Associate Dean, Faculty of Science, School of Computer Science, MIT-WPU, Pune, Maharashtra, India[2]

***Abstract*—Digital systems in the connected world of today bring convenience but also complicated cyber security challenges. The inadequacies of conventional intrusion detection techniques are exposed by the constant adaptation and exploitation of vulnerabilities by advanced cyber threats. Identifying dangers in massive data flows gets more difficult as networks grow, necessitating innovative methods. With the aim of minimizing these concerns, a new ID model is created utilizing cutting-edge machine learning to proactively and flexibly combat dynamic cyber attacks, with regard to evolving cyber attackers, this model seeks to improve accuracy and protection systems. This research develops an arachnid swarm optimization-based Convolutional neural network (ASO opt CNN) model to improve ID performance. An improved modified residual CNN is employed in the model to lessen the vanishing and exploding gradient problems in deep networks and facilitates the optimization process, making it easier for deep networks to learn. The developed model is adjusted using arachnid swarm optimization (ASO), which is the hybridization particle swarm optimization (PSO) and social spider optimization (SSO). Utilizing test data, the model's efficacy is evaluated at last. This test data is also subjected to preprocessing, which leads to the creation of a robust detection model that can identify the presence of network attacks. Experimentation and comparison indicate the approach's effectiveness by attaining accuracies of 95.95%, 95.61%, and 95.00% for three datasets respectively. This highlights the developed model's potential to detect intrusions more effectively.**

Keywords*—Intrusion Detection; arachnid swarm optimization; Convolutional Neural Network; pre-processing; arachnid swarm optimization

I. Introduction

The security of computer networks and systems is of utmost importance in today's technologically advanced and interconnected society. Because vital operations depend more and more on digital infrastructure, there are more sophisticated and varied potential threats from unauthorized access, hostile behavior, and cyber attacks [1]-[3]. By continually monitoring system behavior and network traffic in order to spot suspicious or malicious activity, ID systems serve a critical role in protecting these environments. An essential part of IDS is an ID model, which is in charge of data analysis, pattern recognition, and differentiating between normal and suspicious activity [4]. It works relentlessly to protect the availability, confidentiality, and integrity of digital assets as a vigilant sentinel. To determine potential risks in real-time, this model uses cutting-edge methods from the fields of machine learning and artificial intelligence [5]. Finding unauthorized or hostile actions that can jeopardize the security of a network or system is the main goal of an ID model. The model seeks to identify intrusion attempts, data breaches, and other security breaches by examining network traffic, system logs, and other pertinent data. The model is intended to discover and comprehend the fundamental behavior of a network or system. Then, it continuously scans for departures from this norm, highlighting any actions that are unique or unexpected from a statistical perspective [6]. This method makes it possible to find new, undiscovered hazards. Even though these patterns are not immediately apparent to human analysts, intruders frequently leave behind recognizable patterns in their behavior. The ID approach is able to spot these minute trends and connect seemingly unconnected occurrences to pinpoint potential threats [7]-[9]. The ID methodology functions in real-time, enabling quick reactions to newly emerging threats in a digital environment that is always evolving. The methodology aids in risk mitigation by quickly identifying and warning security personnel about suspicious actions before they worsen [10]. An efficient ID model continuously picks up new information and modifies its detection tactics.

It changes along with the threat environment, ensuring that it continues to be effective against both well-known and new attack vectors. The fundamental elements of an ID model include the model's ability to receive information from a variety of sources, including network traffic, system logs, and application behavior. The analysis is built on top of this data. The characteristics of network traffic and system activity are represented by relevant features or attributes that are taken from the acquired data [11]. The detecting methods use these features as input. The model creates warnings or notifications to notify security administrators or automated systems about potential threats when it notices behaviors that differ from the expected norm. The ID model can start automated responses based on the seriousness of the threat it has discovered or it can suggest human intervention [12]. These reactions could entail isolating affected systems, blocking suspect IP addresses, or modifying network settings. In conclusion, an ID model is a skilled and perceptive keeper who constantly scans digital environments for indicators of harmful intent [13]. Leveraging state-of-the-art machine learning algorithms, this technology enhances the security stance of networks and systems, empowering enterprises to swiftly detect, confront, and mitigate cyber security threats with precision and efficiency. As technology develops, ID models play a more and bigger role in protecting our digital world [14].

Machine learning's contributions to ID significantly enhance both the effectiveness and efficiency of identifying and mitigating cyber security threats. Key advantages include the ability to detect complex and constantly evolving infiltration tactics, a task that can pose challenges for rule-based systems but is well-suited for machine learning algorithms [15]. They are capable of picking up abnormalities and new attack patterns that typical rule sets do not explicitly identify. In order to stay up with new cyber security threats and attack vectors, machine learning models can adapt to new data and learn from it. This versatility guarantees that the ID system will continue to be effective against new attack methods [16].

The number of false positive alerts can be decreased by using machine learning models to thoroughly analyze data patterns. These models can determine whether an activity is indeed suspicious or symptomatic of an intrusion by taking into account a number of different characteristics and contexts [17] [18]. Machine learning algorithms can discover complex correlations between features, improving the accuracy of both known and undiscovered infiltration patterns. As a result, fewer threats are missed and detection rates are improved. Many machine-learning models can analyze data in real time, which enables quick detection and reaction to incursions as they happen [19]. This is essential for reducing possible harm and the effects of attacks. Modern computer systems create a lot of network traffic and system logs, which makes machine-learning methods an ideal choice for this task [20].

The main aim of the research is to develop an ASO opt CNN model to improve ID performance. Getting an intrusion dataset and applying class labels constitute the first phase. Then a model is trained using this labeled dataset. The data is then cleaned and prepared during the preprocessing phase. An updated feature matrix is created once statistical features are extracted from the preprocessed dataset. An improved modified residual Convolutional neural network is fed with the retrieved features. The model is tuned using stages of PSO and SSO. Utilizing test data, the model's efficacy is evaluated in the final phase. This test data is also subjected to preprocessing, which leads to the creation of a robust detection model that can identify the presence of network attacks.

- Arachnid swarm optimization: The hybridization of SSO and PSO seeks to develop an ASO model that takes advantage of each algorithm's strengths while adjusting for its drawbacks. SSO and PSO merger could imply integrating their update methods, sharing tactics, and search strategies to form a new hybrid algorithm. SSO's social sharing mechanism may be included in PSO's velocity update equation, allowing particles to communicate and share information in the same way as social spiders weave webs. Alternatively, to achieve greater convergence to optimal solutions, the exploration and exploitation capacities of both algorithms could be balanced.
- ASO opt CNN: Combining PSO and SSO results in an effective and efficient optimization strategy for CNNs in ID by combining their respective strengths in global exploration and local fine-tuning. Through the use of a hybrid strategy, CNN performance, dynamic threat adaption, and overall ID accuracy may all be improved.

The manuscript maintains its current organizational structure, with Section II providing a comprehensive review of recent research, including methodologies and challenges. Section III presents an illustrative example of an ID model. In Section IV, the novel ASO is introduced. Section V delves into a detailed discussion of experimental results, while Section VI delivers the concluding remarks and summary.

II. Motivation

The growing threat environment of cyber attacks in our linked digital world is what motivated researchers to create an ID model. The sophistication and diversity of cyber dangers increase as technology develops and our reliance on computer networks increases. Hence, a substantial demand exists for robust and adaptable ID systems. This section elucidates the methodologies employed by researchers to enhance the efficacy of ID models.

A. Literature Review

Jie Gua and Shan Lu [21] devised an efficient ID framework that leverages Support Vector Machines (SVM) combined with Naive Bayes feature embedding. Our method performs admirably, delivering excellent accuracy across many datasets. The naive Bayes feature transformation technique, however, may impose some level of computational overhead, which could affect real-time processing efficiency in high-speed network contexts. This poses a potential restriction for this system.

An innovative two-stage intelligent IDS was created by NevrusKaja et al. [22] to identify and defend against such malicious intrusions. The implementation demonstrates a highly effective IDS that detects attacks with high accuracy while removing false positives and increasing computational effectiveness. The model's reliance on ML algorithms for attack detection and classification, meanwhile, has the potential to create limitations to adversarial attacks, thus reducing the system's robustness in the face of sophisticated attackers.

The shortcomings of conventional feature-based ID systems for detecting advanced threat attacks were addressed by Xianwei Gao et al. [23], it debuted a model of adaptive ensemble learning. The proposed solution outperformed current approaches by achieving high accuracy through adaptive. However, the extra computational complexity brought on by ensemble learning could be one of the drawbacks.

SoosanNaderi Mighan1 and Mohsen Kahani [24] developed a hybrid approach aimed at establishing a rapid and highly efficient cyber security ID system. While this method exhibited impressive performance in terms of accuracy, f-measure, sensitivity, precision, and execution time, it's worth noting that the complexity associated with configuring and fine-tuning the hybrid model could potentially introduce certain drawbacks.

An adaptable and robust network ID was created byLirimAshiku and CihanDagli [25] using the deep learning architecture to recognize and categorize network threats. By enabling an adaptive ID system that learns to recognize both known and new network attack patterns, this paradigm improved network security by reducing the chance of intrusion but implementing this model required significant computational resources and skill due to the potential for complex model training and false positives/negatives.

The goal of Mohammad Noor Injadat et al. [26] was to improve network ID with a unique machine learning (ML)--based framework. This model delivered increased detection performance with lower computational complexity, maximizing security measures for people and businesses in the face of rising cyber threats. To achieve the best results, however, the application of several strategies, such as feature selection and oversampling, may create additional complexity.

Peilun Wu and Hui Guo [27] introduced LuNet, a unique hierarchical CNN+RNN neural network that successfully extracts geographical and temporal information from network traffic data, with the goal of improving network ID. Beyond incorporating cutting-edge methodologies, our model excels in the precise and comprehensive identification of networks. It achieves this by adeptly capturing not only the spatial but also the temporal characteristics inherent in network traffic data. However, the deployment and training of LuNet may need significant computational resources, which could lengthen processing times.

The limitations of conventional algorithms were addressed by Yihan Xiao [28] hence creating a network ID model based on CNN-IDS that concentrated on better feature extraction, accuracy, and timely identification of network attacks, which improved the accuracy, decreased false alarm rate, and boosted the timelines. However, the transformation of traffic data into an image format can complicate preparation and might restrict the model's applicability to particular data sets.

B. *Challenges*

- It can be difficult to preprocess network traffic data before feeding it into a CNN model. It may be necessary to use careful engineering to transform raw data into a usable format, which could increase processing overhead.
- ID datasets frequently contain unbalanced class distributions, with normal traffic greatly outnumbering attack cases. It takes sophisticated strategies to train a model to handle such imbalances in order to avoid bias against the dominant class.
- Developing a successful feature extraction plan for the architecture is essential. Despite the fact that CNNs are efficient at learning hierarchical features, it is still difficult to pinpoint the characteristics that are the most useful for ID.
- Finding a balance between underfitting and overfitting can be difficult and time-consuming when optimizing hyperparameters.
- Striking an equilibrium between reducing false positives and minimizing false negatives presents a formidable challenge, as often, mitigating one tends to elevate the potential of the other.

III. Efficient ID Methodology for Arachnid Swarm-Tuned CNN Model

The primary aim of the research is to develop an ASO opt CNN model to improve ID performance. Getting an intrusion dataset and applying class labels constitute the first phase. Then a model is trained using this labeled dataset. The data is then cleaned and prepared during the preprocessing phase. An updated feature matrix is created once statistical features are extracted from the preprocessed dataset. An improved modified residual Convolutional neural network is fed with the retrieved features. The model is tuned using stages of PSO and SSO. Network traffic can be more accurately classified as normal or intrusive by tuning by optimizing the hyperparameters, architecture, and training parameters of the CNN. The model's performance in classifying data can be enhanced by fine-tuning to better capture complex patterns. Utilizing test data, the model's efficacy is evaluated in the final phase. This test data is also subjected to preprocessing, which leads to the creation of a robust detection model that can identify the presence of network attacks. Experimentation and comparison indicate the approach's effectiveness and highlight its potential to considerably increase ID accuracy. The architecture of the developed ID model is illustrated in Fig. 1.

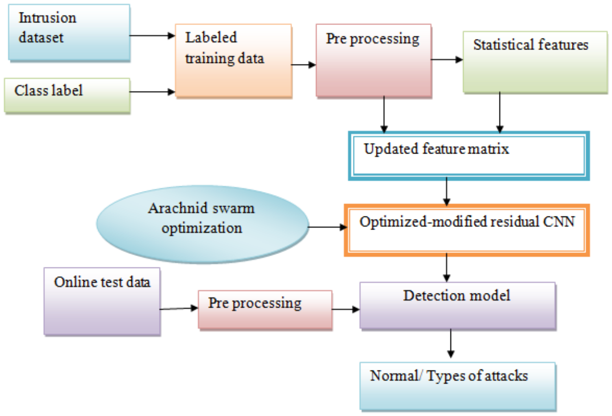


Fig. 1. Architecture of the proposed ID model.

A. *Input*

The inputs for the ID model are gathered from BoT-IoT (d1), CICIDS2017 (d2), and UNSW-NB15 network (d3), which is logically described as follows,

$$K = K_1 + K_2 + K_3 \tag{1}$$

$$K = \sum_{h=1}^{m} K_h + \sum_{i=1}^{u} K_i + \sum_{t=1}^{j} K_t \tag{2}$$

The first dataset, K_h is described as having values between 1 to m, the second dataset, K_i is described as having values between 1 and u, and the third dataset, K_i is described as having values between 1 and j.

B. Data Labeling

After the dataset has been compiled, each data point needs to be assigned a class that describes its nature. These groups in ID generally comprise subcategories like Normal and other kinds of Attacks (such as DoS assaults, malware, and intrusions). Each data point must be labeled according to its behavior, whether it represents a secure network activity or an attack. A crucial stage in machine learning is using the labeled dataset to train a model. The primary aim is to instruct the model in discerning intricate patterns and meaningful correlations between input data, comprising various features, and their respective class labels. This is done in the context of ID by training a model to differentiate between typical network activity and other kinds of attacks.

C. Pre processing

Before being used for training or testing an ID system, network traffic data must first go through a number of data preprocessing procedures in ID. With the help of these procedures, the data is properly structured, cleaned, and modified to improve the functionality of the detection model. In order for the machine learning model to effectively learn from the network traffic data and generalize, data preprocessing is essential in ID. These procedures help the model to more precisely identify and categorize network attacks while reducing false positives and false negatives.

D. Feature Extraction

Feature extraction transforms raw network data into key statistical attributes, creating a streamlined feature matrix. This matrix captures data patterns efficiently, aiding the model in understanding relevant information. Statistical features like mean, variance, standard deviation, skewness, kurtosis, min and max summarize data characteristics. This enhances the model's ability to spot anomalies and patterns, enabling effective ID. By reducing dimensionality and noise, feature extraction optimizes model performance and accuracy.

1) Statistical features: In ID, statistical features are generated numerical metrics from network traffic data characteristics. These properties and behaviors inside network communication are described statistically by these features. The ability to recognize patterns, trends, and abnormalities that may be signs of network attacks depends heavily on statistical aspects. They serve as the foundation for creating efficient ID models. Here are a few typical statistical characteristics used in ID:

a) Mean: An attribute's average value across a range of data points, for instance, the average size of a packet or the average length of a network session. The summing up all of the values for a specific characteristic in a dataset and dividing by the total number of data points, the mean (average) of that attribute is determined. The computation of the mean (μ) of a set of values $(t_1, t_2, ...t_g)$ is shown below in mathematical notation:

$$\mu = (t_1 + t_2 + ...t_g)/g \tag{3}$$

μ is the attribute's mean (average), $t1, t_2$ *and* t_g denotes the attribute's individual values, and g denotes the overall number of data points.

b) Variance: The statistical concept of variance serves as a measure that quantifies the extent or dispersion of data points relative to their mean. In other words, it shows how much a particular data point deviates from the mean (average). Variance is computed by calculating the average of the squared deviations between each data point and the mean. The variance (σ^2) of a group of values $(t_1, t_2, t_3, ...t_g)$ is calculated as follows in mathematical notation:

$$\sigma^2 = \sum(t_f - \mu^2)/g \tag{4}$$

Each unique value of the attribute is represented by t_f, and σ^2 indicates the variance of the values.

c) Standard Deviation (SD): The average departure of the data points from the mean is measured by the SD, which is the variance's square root and is easier to understand. A higher standard deviation denotes more data variability. The standard deviation (σ) is computed mathematically by taking the square root of the variance:

$$\sigma = \sqrt{\sigma^2} \tag{5}$$

The variance and standard deviation of network traffic parameters, such as packet sizes, inter-arrival periods, or payload sizes, are calculated in ID to assist in identifying the typical range of behaviors. High standard deviation values can be a sign of aberrant activity or potential network attacks, which improves the ability of ID systems to detect deviations from the expected variability.

d) Skewness: Skewness, in the context of ID and network traffic data, is a statistical metric that quantifies the asymmetry present in the probability distribution of a real-valued property. The degree to which the distribution is skewed to one side or the other is indicated by its skewness. While negative skewness implies a larger tail on the left side of the distribution, positive skewness suggests a longer tail on the right. The equation and variables that consider the skewness in ID are as follows: A distribution's third standardized moment is measured by skewness. The formula used to calculate it is as follows:

$$\gamma 1 = \left[\sum(t_f - \mu)^3 / (g * \sigma^3)\right] \tag{6}$$

where, t_f stands for each unique value of the attribute, σ for the values' standard deviation, and $\gamma 1$ denotes the skewness of the data.

e) Kurtosis: Kurtosis is a statistical measure that assesses the "tailedness" of the probability distribution of a real-valued property in network traffic data with regard to ID. The concentration of data points in the distribution's tails is revealed by kurtosis. High kurtosis suggests that the data may contain more outliers and heavier tails. The equation and explanation of the variable for kurtosis in ID are given below. The fourth standardized moment of a distribution is measured by kurtosis. This formula is used to compute it:

$$\kappa = \left[\sum \left(t_f - \mu\right)^4 / \left(g * \sigma^4\right)\right] \tag{7}$$

In this case, κ stands for the kurtosis values.

f) Min: The minimal value detected for a particular property within a batch of network traffic data is referred to as the min statistical feature in ID. With the help of this capability, you can understand even the most minute instances of a certain behavior or trait in network communication. Here is a description of the min statistical characteristic in terms of ID. The "min" statistical feature can be written mathematically as:

$$\min = \min\left(t_f\right) \tag{8}$$

where, $\min$ is the attribute's minimal value and t_f stands for each of the attribute's unique values.

g) Max: An attribute's maximum value inside a dataset of network traffic data is referred to as the maximum statistical feature in ID. A high frequency of a particular behavior or trait in network communication is disclosed by this attribute. The max statistical attribute is described in the context of ID in the following sections. The statistical feature known as max has the following mathematical expression:

$$\max = \max\left(t_f\right) \tag{9}$$

where, t_f stands in for each unique value of the attribute and $\max$ is the attribute's maximum value.

E. Updated Feature Matrix

A structured data representation of data comprising features that were taken directly from a dataset is the updated feature matrix. The statistical features received from network traffic data are arranged to create this matrix in the context of ID. These features record pertinent trends, traits, and information about network behaviors that might aid in differentiating between typical usage and potential harmful attacks. The updated feature matrix is produced by converting the raw data into a tabular format, where each row is a data sample (such as a network communication session), and each column denotes a particular feature retrieved from that sample. These characteristics could consist of numerous statistical measurements generated from the network traffic data. To input the data into machine learning models, in this case, the CNN, it is crucial to create an orderly feature matrix. This matrix serves as the model's input as it learns and recognizes intricate patterns that point to the presence of network threats. The updated part of the feature matrix probably refers to the fact that the preprocessing and feature extraction processes clean up the initial raw data, making it more suitable for input into the CNN and raising the overall effectiveness and precision of the ID system. The CNN-based ID approach works effectively because it combines accurate preprocessing, feature extraction, and a well-organized feature matrix.

F. Working of Modified Residual CNN in Intrusion Detection

An improved modified residual CNN leverages adaptive features, and residual units to address the limitations of traditional deep networks. Its ability to handle complex patterns and achieve state-of-the-art performance makes it suitable for intrusion detection. The updated feature matrix dimension becomes the input for the modified residual CNN in the intrusion detection process, where it plays a crucial role in identifying and mitigating network security risks. A potent deep learning architecture called the CNN with residual can be used for ID to automatically discover and extract pertinent features from network traffic data by increasing the depth of the network. Here is a thorough explanation of how modified residual CNN detects intrusions:

1) Convolutional layers: Convolutional layers use filters (called kernels) to move across input data and find pertinent patterns. These filters combine nearby data points to perform convolutions by multiplying each element by an element. Convolutions generate feature maps that depict local patterns and spatial hierarchies. The following equation gives a mathematical description of the convolution layer:

$$B_t = c\left(B_{t-1} \otimes K_t + d_t\right) \tag{10}$$

In this context, K_t represents the weight vector associated with the convolution filter at layer t, where B_t denotes the feature map at layer t, with $BC = J$. Additionally, d_t and c correspond to the bias vector and activation function, respectively. It's noteworthy that the Rectified Linear Unit (ReLU) activation function is a commonly employed non-linear function within CNN. One of the distinguishing characteristics of the CNN is its efficiency in parameter utilization. This efficiency stems from the fact that it employs the same weight and bias vectors across its layers, contributing to a reduction in the overall number of parameters compared to traditional neural networks.

2) Pooling layers: Max pooling is a common technique for combining layers to generate smaller feature maps while maintaining the most crucial data and selecting the highest value possible within a pooling window.

3) Residual unit: A residual block consists of standard convolutional layers followed by batch normalization and ReLU activation. The defining feature is the addition of the input to the output of the convolutional layers. This connection helps address the vanishing gradient issue.

4) Activation functions and non-linearity: The model becomes non-linear as a result of activation functions like the ReLU. They aid residual CNN in learning intricate relationships and identifying significant features.

5) Flattening and fully connected layers: Feature maps are flattened into a 1D vector following numerous

Convolutional and pooling layers. Then the dense layer processes the data and returns some values to determine the intrusions. This vector is processed by fully connected layers, which also learn higher-level abstractions and how features interact.

6) Output layer: The output layer is coupled to the final completely connected layer, which contains neurons that represent potential classes (such as normal or attack). This layer is responsible for producing final predictions and the softmax function is applied, which outputs the probabilities of each class for the given input data.

7) Training and optimization: As input data is associated with relevant class labels (such as normal or attack), labeled data is used to train the residual CNN. The model uses methods like backpropagation and gradient descent during training to modify its internal weights and biases in order to reduce prediction error. Incoming network traffic data can be quickly and accurately classified by trained CNNs. The residual CNN can identify a potential intrusion if the output neuron associated with the attack is highly activated. Residual CNNs are effective in identifying spatial and temporal patterns in network traffic because they learn hierarchical data representations well. In addition to improving ID accuracy and flexibility to change attack patterns, their capacity to automatically learn features minimizes the need for manual feature engineering. The architecture of the modified residual CNN model is depicted in Fig. 2.

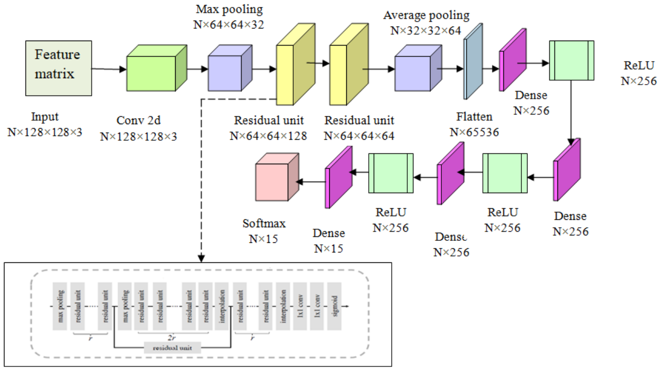


Fig. 2. Architecture of the modified residual CNN model.

G. Testing Phase

In the research, the ASO opt CNN model's effectiveness is thoroughly evaluated using a testing phase. During this phase, the model undergoes evaluation using test data to gauge its precision in discerning between regular network activities and potential intrusion attempts. Before inputting the test data into the model, a preprocessing step is conducted to ensure the data is cleaned and prepared in a consistent manner. The preprocessed test data is then fed into the fine-tuned CNN model, which has been optimized through arachnid swarm optimization. As the test data flows through the model, it generates predictions that indicate whether the network activities are benign or indicative of an attack. This evaluation phase results in the validation and creation of a robust detection model, capable of effectively identifying a wide range of network attacks based on the patterns learned from both training and test data. Through this rigorous experimentation and testing, the research showcases the model's potential to significantly enhance ID accuracy in practical network scenarios.

IV. Proposed Arachnid Swarm Optimization

The utilization of the merging algorithm within the realm of ID serves as a means to enhance and fine-tune the features and parameters of a network-based ID system. The arachnid swarm optimization could tune the residual CNN detection model's weights, thresholds, and hyperparameters by combining the SSO [29] and PSO [30] search algorithms. In order to construct a new hybrid algorithm, SSO and PSO may combine their update methods, sharing strategies, and search strategies. The SSO social sharing mechanism might be implemented into the PSO velocity update equation, allowing particles to communicate and exchange information in the same way that social spiders create webs. Alternatively, to improve convergence to optimal solutions, the exploration and exploitation capabilities of both algorithms could be balanced.

A. Population Initialization

The SSO is a series of repeated procedures that begin by randomly initializing the entire population, and the main feature of social spiders is female-biased populations. In ancient times, the percentage of female S_g was arbitrarily assigned in the domain of percent (65-90) of total population S_u. S_g is so calculated using the following equation:

$$S_v = floor\left[\left(0.9 - rand(0,1) * 0.25\right) * S_u\right] \tag{11}$$

The number of a male spider S_i is calculated as the product of S_v and S_u.

$$S_i = S_u - S_v \tag{12}$$

B. Assignation of Fitness

The size of the spider's ability to properly complete the prescribed duties is the feature that evaluates a personal ability. Each spider in the supplied strategy has a weight t_g, which illuminates the fineness solution that meets with the population D of spider g. The following formulas are used to determine the fitness of each spider.

$$t_g = \frac{K(D_g) - worst_D}{best_D - worst_D} \tag{13}$$

where, $K(D_g)$ is the fitness value obtained from the spider location D_g assessment.

The following equation can be used to calculate the value of the worst and best solution:

$$best_D = \max(K(D_m)) \qquad m \in \{1,2,\ldots S\} \tag{14}$$

$$worst_D = \min(K(D_m)) \qquad m \in \{1,2,\ldots S\} \tag{15}$$

C. Modeling of the Vibrations

The communal web acts as a conduit for communication among colony members, facilitating the exchange of vital information. Vibrations, denoted as (Egi), are influenced by both the weight and the distance of the spider responsible for their creation. These vibrations represent the outcome of information transmitted by a member, denoted as b and are meticulously modeled using the following equation, involving an individual's contribution.

$$Egi_{g,b} = t_b.d^{-e_{g,b}^2} \tag{16}$$

The formula computed the distance between spiders g and b.

$$e_{g,b} = \left\| D_g - D_b \right\| \tag{17}$$

The SSO method took into account three distinct correlations (three vibrations):

1) Egj_{fg} Vibration: Created by individual $g(D_g)$ in response to the transmission of information provided by member $f(D_g)$, f is the closest member to g and has a higher weight t_f :

$$Egi_{fg} = t_f.d^{-e_{g,m}^2} \tag{18}$$

2) Egj_{jg} Vibration: Created by the individual $g(D_g)$ in response to the transmission of information provided by member $j(D_j)$, where j is the member with the highest weight t_j :

$$Egi_{jg} = t_j.d^{-e_{g,j}^2} \tag{19}$$

3) Egj_{vg} Vibration: Created by the individual $g(D_g)$ as an outcome of information provided by member $g(D_v)$, where v is the closest female member to g :

$$Egi_{vg} = t_v.d^{-e_{g,v}^2} \tag{20}$$

D. Population Initializing

The first phase is an iterative procedure similar to previous SSO evolutionary algorithms in which the entire population (males and females) is randomly started, beginning by initializing the set D of S social-spider positions. Each $spiderv_g or\, i_g$ location is a q dimensional vector containing the parameter values that need to be improved. These parameter values are divided between the original parameter r_b^{high} specified upper limit and the preliminary parameter r_b^{low} lower limit. The equations following describe this:

$$v_{g,b}^y = r_b^{low} + rand(0,1) * \left(r_b^{high} - r_b^{low} \right) \qquad g = 1,2,\ldots S_v \text{ and } b = 1,2,..q \tag{21}$$

$$i_{g,b}^y = r_b^{low} + rand(0,1) * \left(r_b^{high} - r_b^{low} \right) \qquad g = 1,2,\ldots S_i \text{ and } b = 1,2,..q \tag{22}$$

Individual indices are indicated by b and g while the function $(rand(0,1))$ generates a random number spanning from 0 to 1. The initial population is denoted as "zero."

E. Cooperative Operators

Spiders' cooperative behavior is determined by their gender as well as other elements such as curiosity, reproductive cycle, and other random phenomena.

1) Integrating phase: The hybridization of SSO and PSO seeks to take advantage of each algorithm's strengths while adjusting for its drawbacks. SSO and PSO merger could imply integrating their update methods, sharing tactics, or search strategies to form a new hybrid algorithm. SSO's social sharing mechanism is included in PSO's velocity update equation, allowing particles to communicate and share information in the same way as social spiders weave webs. Alternatively, to achieve greater convergence to optimal solutions, the exploration and exploitation capacities of both algorithms could be balanced.

2) Female cooperative: The following examples illustrate how female spiders may attract or repel other spiders:

$$A = 0.5v_g^{m+1} + 0.5q_{je}^{m+1} \tag{23}$$

$$A=0.5\left(\begin{cases} v_g^m + \delta.vib_{mg}.\left(D_m - v_g^m\right) + \varepsilon.vib_g.\left(D_m - v_g^m\right) \\ \quad +\gamma.(rand - 0.5)\, with\ probability\, rZ \\ v_g^m - \delta.vib_{mg}.\left(D_m - v_g^m\right) - \varepsilon.vib_g.\left(D_m - v_g^m\right) + \\ \gamma.(rand - 0.5) with\ probability 1 - rZ \end{cases} q_{je}^m + s_1 u_1\left(b_{je}^m - r_{je}^m\right) + s_2 u_2\left(b_{ie}^m - r_{je}^m\right) \right) \tag{24}$$

In this context, m signifies the number of iterations, while $\delta, \varepsilon, \gamma$ represent random values falling within the range of $[0,1]$. The individuals D_m and D_j stand for the best individual in the entire D population and the closest member to g with the highest weight.

In the given equation, $e = 1,2,\ldots.E$ represents the dimension, and $j = 1,2,..T$ represents the particle index within the swarm. T denotes the swarm size, while $s1$ and $s2$ are constants referred to as cognitive and social scaling parameters, sometimes known as acceleration coefficients. u_1, u_2 are the random numbers drawn from a uniform distribution in the range $[0,1]$. Equation 24 highlights that each dimension of every particle is updated independently of the others. The sole connection between these dimensions in the problem space is established through the objective function, which relies on the best positions discovered thus far, denoted as $jbest$ and $bbest$.

3) Male cooperative: Male spiders are categorized into two groups, namely dominant and non-dominant, based on their respective weights. To benefit from the resources that are being squandered by the dominant spiders, the non-dominant individuals are drawn to the weighted mean of the male population. The update of the male spider positions can therefore be stated as follows:

$$B = 0.5 i_g^{m+1} + 0.5 q_{je}^{m+1} \tag{25}$$

$$B = 0.5 \left(\begin{cases} i_g^m + \delta . vib_g . \left(D_v - i_g^m \right) + \rho . \left(rand - 0.5 \right) \\ when\ T_{S_{v+g}} > T_{S_{v+g}} \\ i_g^m + \delta . \left(\dfrac{\sum_{x-1}^{ci} i_x^m . TS_{v+x}}{\sum_{x-1}^{ci} TS_{v+x}} \right) \\ when\ T_{S_{v+g}} \le T_{S_{v+i}} \end{cases} + q_{je}^m + s_1 u_1 \left(b_{je}^m - r_{je}^m \right) + s_2 u_2 \left(b_{ie}^m - r_{je}^m \right) \right) \tag{26}$$

The closest female member to the male member g is represented by the individual D_v, while the weighted mean of the male population F is represented by the individual $\left(\dfrac{\sum_{x-1}^{ci} i_x^m . TS_{v+x}}{\sum_{x-1}^{ci} TS_{v+x}} \right)$.

Algorithm 1: Pseudo code for the proposed arachnid swarm optimization

S.NO	Pseudo code for the proposed arachnid swarm optimization
1	Initialize swarm population for SSO
2	Initialize swarm population for PSO
3	Initialize the best solution
4	Initialize max iterations
5	Initialize iteration counter
6	while iteration counter < max iterations:
7	Evaluate the fitness of SSO and PSO populations
8	Update female spiders' positions and vibrations using SSO
9	Update male spiders' positions using SSO
10	Calculate the best solution found by SSO
11	Update particles' velocities and positions using PSO
12	Evaluate the fitness of the PSO population
13	Calculate the best solution found by PSO
14	if fitness(SSO best solution) > fitness(PSO best solution):
15	Combined best solution = SSO best solution
16	else:
17	Combined best solution = PSO best solution
18	if fitness(combined best solution) > fitness(best solution):
19	Best solution = combined best solution
20	Iteration counter += 1
21	Output best solution

V. Result and Discussion

An ID model is meticulously developed using the ASO-optimized CNN, and its effectiveness is rigorously evaluated in comparison to alternative methodologies.

A. Experimental Setup

ID is conducted using the Python programming language with the Windows 10 operating system.

B. Dataset Description

D1 [31]: This dataset is a crucial asset in the realm of ID for Internet of Things (IoT) environments. It offers a diverse range of network traffic data, encompassing both normal communication patterns among various IoT devices and simulated malicious activities, including botnet-related behaviors. Researchers utilize this dataset to develop and evaluate ID systems and security solutions specifically tailored to the unique challenges posed by IoT networks. It serves as a fundamental resource for enhancing the cyber security of IoT ecosystems by facilitating the identification and mitigation of potential threats and anomalies.

D2 [32]: This dataset is a significant asset in the field of ID. It consists of a diverse range of network traffic data, including both benign and malicious activities. This dataset is instrumental in the development and evaluation of intrusion detection systems and cyber security solutions. Researchers leverage CICIDS2017 to enhance the security of networks by effectively identifying and mitigating potential threats and anomalies in a controlled, real-world environment.

D3 [33]: It offers a comprehensive collection of network traffic data, featuring a variety of benign network activities and simulated cyber-attacks. This dataset facilitates the development, testing, and evaluation of intrusion detection systems and security mechanisms. Researchers and cyber security experts rely on UNSW-NB15 to enhance network security by efficiently identifying and countering potential threats and anomalies.

C. Comparative Methods

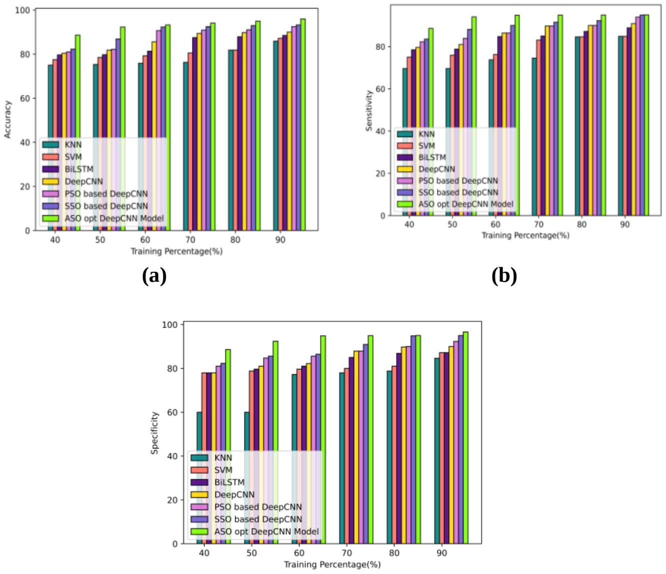
The ASO opt CNN model undergoes an evaluation where it is compared to various existing models. These existing include KNN [34], SVM [35], BiLSTM [36], deep CNN [37], PSO-based deep CNN [38], and SSO-based deep CNN [39] to gauge its performance.

1) Comparative analysis based on TP for d1: Fig. 3 illustrates the TP 90 metrics, used to compare the efficacy of the ASO-optimized CNN with other comparative techniques.

Fig. 3(a) depicts the ASO-optimized CNN model's ID accuracy. The ASO opt CNN achieves a remarkable accuracy of 95.95% with a TP of 90, outperforming the SSO-based BiLSTM by 2.84%.

Fig. 3(b) showcases the ID sensitivity of the ASO-optimized CNN model. With a TP of 90, the ASO opt CNN demonstrates a remarkable sensitivity of 95.00%, surpassing the SSO-based BiLSTM by 0.05%.

In Fig. 3(c), the ID specificity of the ASO-optimized CNN model is displayed. Achieving a TP of 90, the ASO opt CNN exhibits a specificity of 96.61%, outperforming the SSO-based BiLSTM by a margin of 1.67%.



(a) (b)

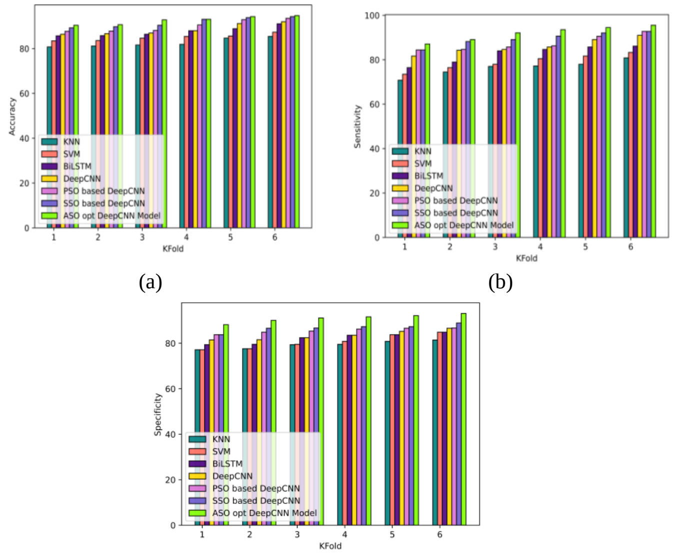
c)

Fig. 3. Comparative analysis concerning TP a) accuracy, b) sensitivity, and c) specificity.

2) Comparative analysis based on K-fold for d1: Fig. 4(a) depicts the ASO-optimized CNN model's ID accuracy. The ASO opt CNN achieves a remarkable accuracy of 94.71% with a k-fold 6, outperforming the SSO-based BiLSTM by 0.47%.

Fig. 4(b) showcases the ID sensitivity of the ASO-optimized CNN model. With a k-fold 6, the ASO opt CNN demonstrates a remarkable sensitivity of 95.55%, surpassing the SSO-based BiLSTM by 2.93%.

In Fig. 4(c), the ID specificity of the ASO-optimized CNN model is displayed. Achieving a k-fold 6, the ASO opt CNN exhibits a specificity of 93.12%, outperforming the SSO-based BiLSTM by a margin of 4.51%.



(a) (b)

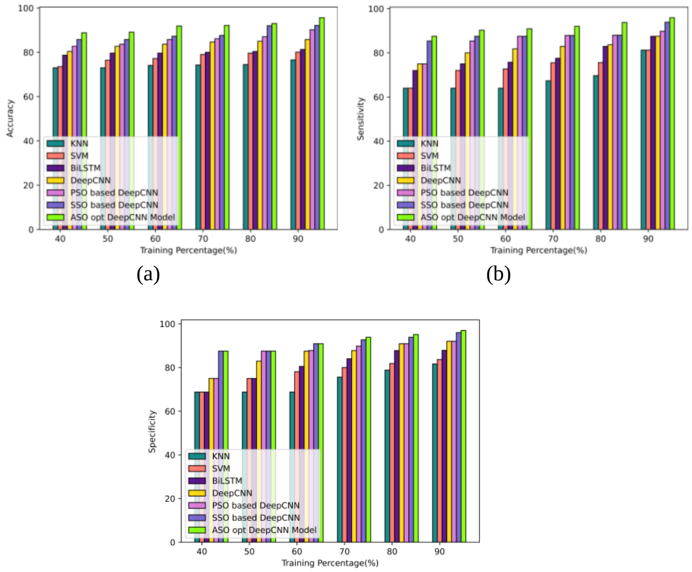
c)

Fig. 4. Comparative analysis concerning K-fold a) accuracy, b) sensitivity, and c) specificity.

3) Comparative analysis based on TP for d2: Fig. 5(a) depicts the ASO-optimized CNN model's ID accuracy. The ASO opt CNN achieves a remarkable accuracy of 95.61% with a TP of 90, outperforming the SSO-based BiLSTM by 3.65%.

Fig. 5(b) showcases the ID sensitivity of the ASO-optimized CNN model. With a TP of 90, the ASO opt CNN demonstrates a remarkable sensitivity of 95.92%, surpassing the SSO-based BiLSTM by 2.13%.

In Fig. 5(c), the ID specificity of the ASO-optimized CNN model is displayed. Achieving a TP of 90, the ASO opt CNN exhibits a specificity of 96.96%, outperforming the SSO-based BiLSTM by a margin of 0.99%.



(a) (b)

c)

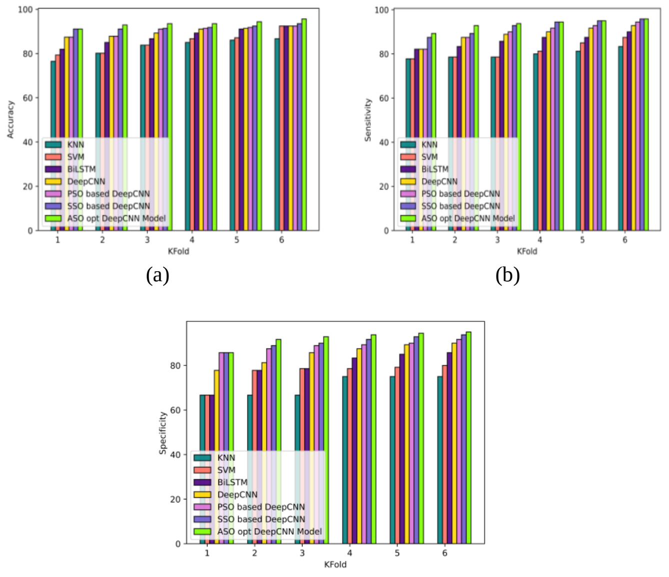
Fig. 5. Comparative analysis concerning TP a) accuracy, b) sensitivity, and c) specificity.

4) Comparative analysis based on K-fold for d2: Fig. 6(a) depicts the ASO-optimized CNN model's ID accuracy. The ASO opt CNN achieves a remarkable accuracy of 95.63% with a k-fold 6, outperforming the SSO-based BiLSTM by 2.22%.

Fig. 6(b) showcases the ID sensitivity of the ASO-optimized CNN model. With a k-fold 6, the ASO opt CNN demonstrates a remarkable sensitivity of 95.55%, surpassing the SSO-based BiLSTM by 2.93%.

In Fig. 6(c), the ID specificity of the ASO-optimized CNN model is displayed. Achieving a k-fold 6, the ASO opt CNN exhibits a specificity of 95.00%, outperforming the SSO-based BiLSTM by a margin of 1.32%.

5) Comparative analysis based on TP for d3: Fig. 7(a) depicts the ASO-optimized CNN model's ID accuracy. The ASO opt CNN achieves a remarkable accuracy of 95.00% with a TP of 90, outperforming the SSO-based BiLSTM by 3.51%.



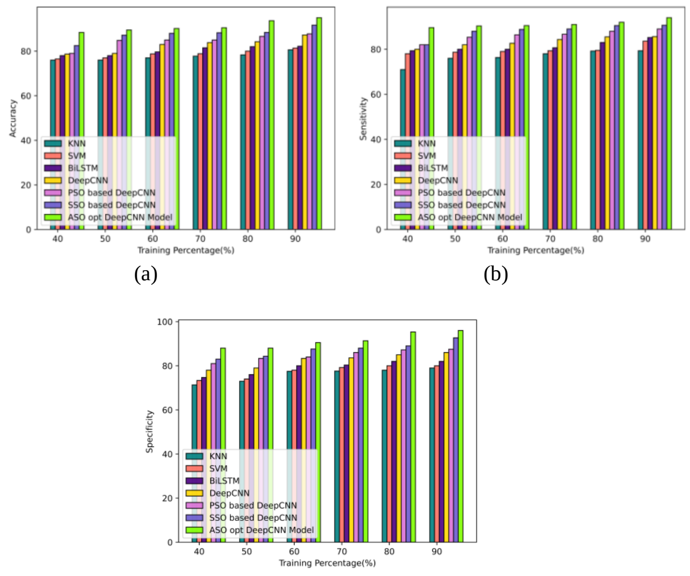
(a) (b)

c)

Fig. 6. Comparative analysis concerning K-fold a) accuracy, b) sensitivity, and c) specificity.

Fig. 7(b) showcases the ID sensitivity of the ASO-optimized CNN model. With a TP of 90, the ASO opt CNN demonstrates a remarkable sensitivity of 94.00%, surpassing the SSO-based BiLSTM by 3.55%.

In Fig. 7(c), the ID specificity of the ASO-optimized CNN model is displayed. Achieving a TP of 90, the ASO opt CNN exhibits a specificity of 96.00%, outperforming the SSO-based BiLSTM by a margin of 3.47%.



(a) (b)

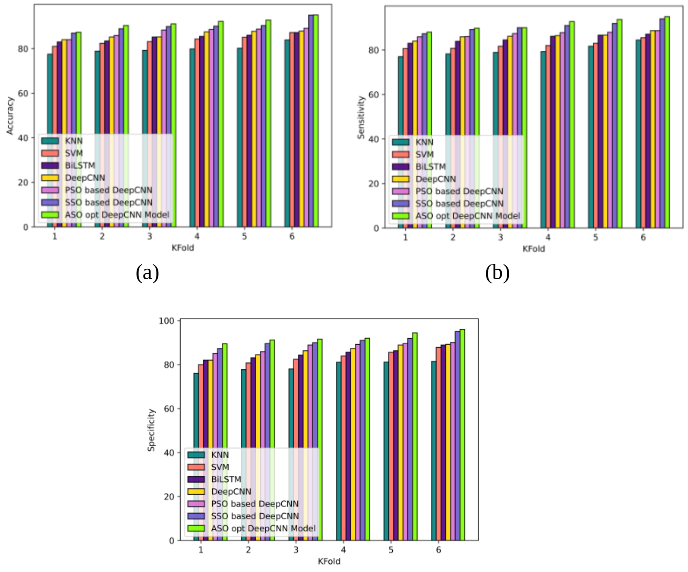
c)

Fig. 7. Comparative analysis concerning TP a) accuracy, b) sensitivity, and c) specificity.

6) Comparative analysis based on K-fold for d3: Fig. 8(a) depicts the ASO-optimized CNN model's ID accuracy. The ASO opt CNN achieves a remarkable accuracy of 95.09% with a k-fold 6, outperforming the SSO-based BiLSTM by 0.10%.

Fig. 8(b) showcases the ID sensitivity of the ASO-optimized CNN model. With a k-fold 6, the ASO opt CNN demonstrates a remarkable sensitivity of 95.00%, surpassing the SSO-based BiLSTM by 1.05%.

In Fig. 8(c), the ID specificity of the ASO-optimized CNN model is displayed. Achieving a k-fold 6, the ASO opt CNN exhibits a specificity of 96.00%, outperforming the SSO-based BiLSTM by a margin of 1.04%.



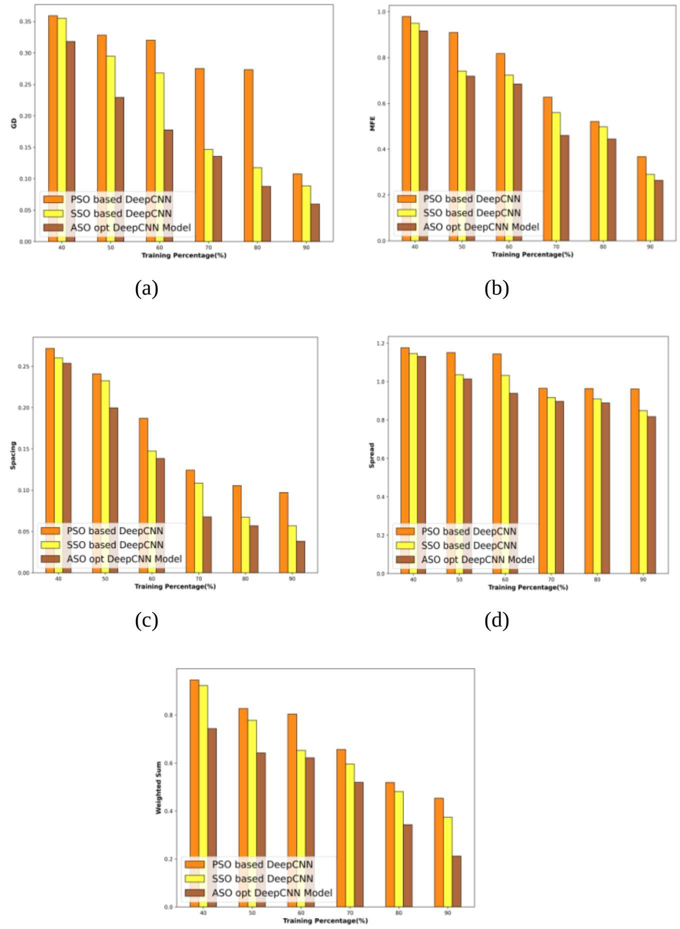
(a) (b)

c)

Fig. 8. Comparative analysis concerning TP a) accuracy, b) sensitivity, and c) specificity.

D. Comparative Analysis Based on Quality Metrics

A comparative evaluation of the proposed method with other existing methods based on quality metrics such as generation distance (GD), Maximum Pareto front error (MFE), Spacing, Spread, and weighted sum is conducted and is presented in Fig. 9 and the results obtained from the analysis in terms of those quality metrics are depicted in Table I. Fig. 9(a) shows that the proposed ASO opt Deep CNN Model attains a low GD of 0.06 showing that the proposed method has the best convergence with the Pareto optimal front. Fig. 9(b) reveals the proposed method attains an MFE of 0.26 which is much less than other compared methods showing the effectiveness of the proposed approach. Fig. 9(c) indicates the spacing metric which is 0.04 for the proposed method revealing that the approach can have a uniform distribution of Pareto points on the curve compared to other approaches. The spread of the proposed model is 0.82 which shows its best distribution and extension of solutions than other methods and is depicted in Fig. 9(d). Fig. 9(e) shows the weighted sum of the proposed approach as 0.21 which is very low and indicates that it is better than other compared approaches.



(a) (b)

(c) (d)

(e)

Fig. 9. Comparative analysis based on Quality Metrics a) GD, b) MFE, c) Spacing, d) Spread, and e) Weighted Sum.

TABLE I. COMPARISON OF PROPOSED ASO OPT DEEP CNN METHOD WITH EXISTING METHODS BASED ON QUALITY METRICS

Models	GD	MFE	Spacing	Spread	Weighted Sum
PSO-based Deep CNN	0.11	0.37	0.10	0.96	0.45
SSO-based Deep CNN	0.09	0.29	0.06	0.85	0.37
ASO opt for Deep CNN Model	0.06	0.26	0.04	0.82	0.21

E. Computational Complexity Analysis

The analysis of the computational complexity of the ASO-opt deep CNN with traditional approaches is presented in Table II. The superiority of the ASO opt Deep CNN Model is demonstrated by comparing it with other methods based on computing time across several iterations. Additionally, the developed method outperforms all other known methods with a low computational time of 20.46 for D1, 20.51 for D2, and 20.13 for D3 at iteration 100. The results highlight the ASO opt Deep CNN technique's computational efficiency by demonstrating that it regularly completes tasks far faster than those of other available techniques.

TABLE II. COMPARISON OF COMPUTATIONAL TIME

Models	Computational time		
	D1	**D2**	**D3**
KNN	20.84	20.84	20.83
SVM	20.56	20.59	20.59
BiLSTM	20.72	20.73	20.77
Deep CNN	20.78	20.74	20.79
PSO-based Deep CNN	20.80	20.81	20.80
SSO-based Deep CNN	20.80	20.83	20.81
ASO opt Deep CNN Model	20.46	20.51	20.13

F. Comparative Discussion

The developed ASO opt Deep CNN Model compares with conventional approaches to prove its effectiveness in intrusion detection. Even though, the existing methods show effective performance in intrusion detection, they still have some limitations, such as KNN [34] being computationally expensive and extracting irrelevant features, which affects the model's performance. Likewise, SVM [35] is also a time and memory-consuming model. BiLSTM [36] struggles with long sequences due to memory constraints and also suffers from gradient issues during training. Deep CNN [37] suffers from overfitting and requires a large amount of data for effective training. PSO-based Deep CNN [38] struggles with high dimensional spaces and SSO-based deep CNN has generalization issues. Therefore, the ASO opt Deep CNN Model is developed here for efficient intrusion detection and the ASO aids in improving the model's performance in intrusion detection by fine-tuning the parameters of the deep CNN. The use of ASO in this model also reduces the risk of overfitting and reduces the time complexity of the model due to its fast convergence. The results show that the conventional methods attain very low accuracy compared to the proposed method. This low accuracy is attained due to the generalization issues, overfitting issues, irrelevant feature extraction, and time complexity issues in existing approaches. Nevertheless, the proposed method solves these existing issues and attained high accuracy in detecting intrusion compared to other existing methods. The ASO opt Deep CNN Model serves as an effective solution for intrusion detection and the model has the ability to handle large-scale network data since it is tested on three large intrusion datasets. Eventhough, though the model requires more computational resources for training large datasets; the use of ASO reduces the need for more computational time, indicating its efficiency in handling large-scale data. The model also has the ability to be implemented in the real world in various network environments. However, real-world data often has imbalanced classes and the complicated attackers may try to evade detection by crafting adversarial examples. These issues can be solved by employing oversampling or under-sampling techniques to address class imbalance issues and the development of robust techniques also helps to enhance the model's resilience, which can be done in the future. Tables III and IV provide a comprehensive comparative analysis of the ASO-opt deep CNN, alongside several other existing approaches. The findings from this analysis clearly demonstrate that the ASO-opt deep CNN excels, surpassing the performance of the other methods examined in the realm of ID.

TABLE III. COMPARATIVE DISCUSSION TABLE FOR TP

Models	TP 90								
	D1			D2			D3		
	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
KNN	85.90	84.85	84.62	76.56	81.25	81.63	80.60	79.33	79.00
SVM	87.18	84.85	87.18	80.07	81.25	83.67	81.40	83.60	80.00
BiLSTM	88.46	88.89	87.18	81.27	87.50	87.88	82.20	85.20	82.00
Deep CNN	90.00	90.91	90.00	85.75	87.50	92.00	87.25	85.60	86.00
PSO based Deep CNN	92.50	93.94	92.31	90.16	89.80	92.00	87.75	89.00	87.50
SSO based Deep CNN	93.22	94.95	95.00	92.12	93.88	96.00	91.67	90.67	92.67
ASO opt Deep CNN Model	95.95	95.00	96.61	95.61	95.92	96.96	95.00	94.00	96.00

TABLE IV. COMPARATIVE DISCUSSION TABLE FOR K-FOLD

Models	K-fold 6								
	D1			D2			D3		
	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity	Accuracy	Sensitivity	Specificity
KNN	85.43	80.85	81.41	86.70	83.33	75.00	83.89	84.44	81.45
SVM	87.32	83.30	84.88	92.44	87.50	80.00	87.22	85.56	87.78
BiLSTM	91.07	86.12	84.88	92.44	90.00	85.71	87.22	87.10	88.89
Deep CNN	91.98	91.00	86.61	92.44	92.86	90.00	87.91	88.71	89.19
PSO based Deep CNN	93.50	92.75	86.71	92.44	94.44	91.67	89.11	88.71	90.10
SSO based Deep CNN	94.27	92.75	88.92	93.50	95.83	93.75	95.00	94.00	95.00
ASO opt Deep CNN Model	94.71	95.55	93.12	95.63	95.83	95.00	95.09	95.00	96.00

VI. CONCLUSION

In summary, this research focuses on improving ID performance through an ASO-opt CNN model. It follows a comprehensive methodology, starting with dataset acquisition and model training, followed by data preprocessing and feature extraction. An enhanced CNN model is introduced and fine-tuned through PSO and SSO optimization stages, enhancing its ability to classify network traffic accurately. The final phase evaluates the model's effectiveness using test data, resulting in a robust detection system. Experiments highlight the approach's efficacy and its potential to significantly boost ID accuracy, making it a valuable asset in the ever-evolving cybersecurity landscape. The ASO-opt CNN model demonstrated outstanding performance in TP 90, achieving high accuracy for d1 95.95%, d2 95.61% and for d3 95.00%, sensitivity of d1 95.00%, d2 95.92% and d3 94.00%, finally specificity of d1 96.61%, d2 96.96% and d3 96.00% for different datasets. In k-fold 6, the model's effectiveness remained strong with impressive accuracy of d1 94.71%, d2 95.63% and d3 95.09%, sensitivity of d1 95.55%, d2 95.83% and d3 95.00%, and finally specificity of d1 93.12%, d2: 95.00% and d3 96.00%. These exceptional results highlight the model's reliability and potential to significantly enhance intrusion detection accuracy. In future, additional hybrid optimization techniques can be employed to improve the model’s performance. Different ensemble classifiers may also be employed for effective performance in intrusion detection. The model can also be improved to make it suitable for detecting other cyber threats.

REFERENCES

[1] Y. Lin, X. Zhu, Z. Zheng, Z. Dou, and R. Zhou, “The individual identification method of wireless device based on dimensionality reduction and machine learning.” The journal of supercomputing, 75(6), pp.3010-3027, 2019.

[2] M. Liu, T. Song, J. Hu, J. Yang, and G. Gui, “Deep learning-inspired message passing algorithm for efficient resource allocation in cognitive radio networks.” IEEE Transactions on Vehicular Technology, 68(1), pp.641-653, 2018.

[3] G. Gui, H. Huang, Y. Song, and H. Sari, ‘‘Deep learning for an effective nonorthogonal multiple access scheme,’’ IEEE Trans. Veh. Technol., vol. 67, no. 9, pp. 8440–8450, Sep. 2018.

[4] H. Huang, J. Yang, H. Huang, Y. Song, and G. Gui, ‘‘Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system,’’ IEEE Trans. Veh. Technol., vol. 67, no. 9, pp. 8549–8560, Sep. 2018.

[5] Y. Li, X. Cheng, and G. Gui, ‘‘Co-robust-ADMM-net: Joint ADMM framework and DNN for robust sparse composite regularization,’’ IEEE Access, vol. 6, pp. 47943–47952, 2018.

[6] Y. Lin, C. Wang, C. Ma, Z. Dou, and X. Ma, ‘‘A new combination method for multisensor conflict information,’’ J. Supercomput., vol. 72, no. 7,pp. 2874–2890, 2016.

[7] F. Kuang, W. Xu, and S. Zhang, ‘‘A novel hybrid KPCA and SVM with GA model for ID,’’ Appl. Soft Comput., vol. 18, pp. 178–184,May 2014.

[8] K. V. Narayana, V. V. R. Manoj, and K. Swathi, ‘‘Enhanced face recognition based on PCA and SVM,’’ Int. J. Comput. Appl., vol. 117, no. 2,pp. 40–42, 2015

[9] C.-F. Tsai, Y.-F. Hsu, C.-Y. Lin, and W.-Y. Lin, “ID by machine learning: A review,” Expert Systems with Applications, vol. 36, no. 10, pp. 11 994–12 000, 2009.

[10] A. Moubayed, M. Injadat, A. Shami, and H. Lutfiyya, “Student engagement level in e learning environment: Clustering using k-means,” American Journal of Distance Education, vol. 34, no. 2, 2019.

[11] M. Injadat, F. Salo, and A. B. Nassif, “Data mining techniques in social media: A survey,” Neurocomputing, vol. 214, pp. 654 – 670, 2016. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S092523121630683X

[12] M. B. Salem, S. Hershkop, and S. J. Stolfo, “A survey of insider attack detection research,” in Insider Attack and Cyber Security. Springer, 2008, pp. 69–90.

[13] I. Ahmad, M. Basheri, M. J. Iqbal, and A. Rahim, “Performance comparison of support vector machine, random forest, and extreme learning machine for ID,” IEEE Access, vol. 6, pp. 33789–33795, 2018.

[14] W. Bul’ajoul, A. James, and M. Pannu, “Improving network ID system performance through quality of service configuration and parallel technology,” Journal of Computer and System Sciences, vol. 81, no. 6, pp. 981–999, 2015.

[15] S. X. Wu and W. Banzhaf, “The use of computational intelligence in ID systems: A review,” Applied soft computing, vol. 10, no. 1, pp. 1–35, 2010.

[16] S. M. H. Bamakan, B. Amiri, M. Mirzabagheri, and Y. Shi, “A new ID approach using pso based multiple criteria linear programming,” Procedia Computer Science, vol. 55, pp. 231–237, 2015.

[17] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, “Network ID,” IEEE Network, vol. 8, no. 3, pp. 26–41, 1994.

[18] R. Sommer and V. Paxson, “Outside the closed world: On using machine learning for network ID,” in 2010 IEEE Symposium on Security and Privacy, pp. 305–316, IEEE, 2010.

[19] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fern ´ andez, and ´ E. Vazquez, “Anomaly-based network ID: Techniques, ´ systems and challenges,” Computers & Security, vol. 28, no. 1-2, pp. 18–28, 2009.

[20] A. L. Buczak and E. Guven, “A survey of data mining and machine learning methods for cyber security ID,” IEEE Communications Surveys & Tutorials, vol. 18, no. 2, pp. 1153–1176, 2015.

[21] R.Vinayakumar, M. Alazab, K.P. Soman, P. Poornachandran, A. Al-Nemrat, and S. Venkatraman, “Deep learning approach for intelligent intrusion detection system.” Ieee Access, 7, pp.41525-41550, 2019.

[22] N. Kaja, S. Adnan and M. Di "An intelligent ID system." Applied Intelligence 49: 3235-3247, 2019.

[23] X. Gao, C. Shan, C. Hu, Z. Niu, and Z. Liu, “An adaptive ensemble machine learning model for intrusion detection.” Ieee Access, 7, pp.82512-82521, 2019.

[24] C. Liu, Z. Gu, and J. Wang, “A hybrid intrusion detection system based on scalable K-means+ random forest and deep learning.” Ieee Access, 9, pp.75729-75740, 2021.

[25] L. Ashiku, and D. Cihan, "Network ID system using deep learning." Procedia Computer Science 185: 239-247, 2021.

[26] M. Injadat, A. Moubayed, A.B. Nassif, and A. Shami, “Multi-stage optimized machine learning framework for network intrusion detection.” IEEE Transactions on Network and Service Management, 18(2), pp.1803-1816, 2020.

[27] Q. Wu, H. Zhang, R. Jing, and Y. Li, “Feature selection based on twin support vector regression.” In 2019 IEEE symposium series on computational intelligence (SSCI), 2903-2907, 2019.

[28] Y. Xiao, X. Cheng Z. Taining and Z. Zhongkai "An ID model based on feature reduction and convolutional neural networks." IEEE Access 7 (2019): 42210-42219.

[29] A. Luque-Chang, E. Cuevas, F. Fausto, and M. Pérez, “Social spider optimization algorithm: modifications, applications, and perspectives.” Mathematical Problems in Engineering, 2018, pp.1-29, 2018.

[30] D. Wang, D. Tan, and L. Liu, “Particle swarm optimization algorithm: an overview.” Soft computing, 22, pp.387-408, 2018.

[31] BOT-IOT dataset: https://research.unsw.edu.au/projects/bot-iot-dataset)

[32] CICIDS2017 dataset: https://www.unb.ca/cic/datasets/ids-2017.html)

[33] UNSW-NB15 network dataset: https://www.kaggle.com/datasets/dhoogla/unswnb15))

[34] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, “KNN model-based approach in classification.” In On The Move to Meaningful Internet Systems 2003: CoopIS, DOA, and ODBASE: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE 2003, Catania, Sicily, Italy, November 3-7, 2003. Proceedings (pp. 986-996). Springer Berlin Heidelberg. 2003.

[35] C. Schuldt, I. Laptev, and B. Caputo, “Recognizing human actions: a local SVM approach.” In Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004. (Vol. 3, pp. 32-36), 2004.

[36] S. Siami-Namini, N. Tavakoli, and A.S. Namin, “The performance of LSTM and BiLSTM in forecasting time series.” In 2019 IEEE International conference on big data (Big Data) (pp. 3285-3292), 2019.

[37] K. Zhang, W. Zuo, S. Gu, and L. Zhang, “Learning deep CNN denoiser prior for image restoration.” In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3929-3938), 2017.

[38] K. S. Bhuvaneshwari, K. Venkatachalam, S. Hubálovský, P. Trojovský, and P. Prabu. "Improved Dragonfly Optimizer for ID Using Deep Clustering CNN-PSO Classifier." Computers, Materials & Continua 70, no. 3 (2022).

[39] K.M. Alalayah, F.S. Alrayes, J.S. Alzahrani, K.M. Alaidarous, I.M. Alwayle, H. Mohsen, I.A. Ahmed, and M. Duhayyim, “Optimal Deep Learning Based Intruder Identification in Industrial Internet of Things Environment.” Comput. Syst. Sci. Eng., 46(3), pp.3121-3139, 2023.