



An Enhanced Hybrid Intrusion Detection Based on Crow Search Analysis Optimizations and Artificial Neural Network

Chetan Gupta¹ · Amit Kumar¹ · Neelesh Kumar Jain¹

Accepted: 20 January 2024 / Published online: 28 February 2024

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

The continuous advancement of computer networks has given rise to grave concerns regarding security and susceptibility. Network administrators utilize intrusion detection systems (IDS) to deliver essential network security. Current IDS devices produce false alarms in response to routine user activities rather than detecting novel assaults. Neural networks may be used to overcome this problem and increase detection accuracy. In this paper, we propose a hybrid approach based on neural networks and correlation-based feature selection to detect anomalies. Experimental research is done on the standard dataset NSL-KDD for intrusion detection using current attacks. We introduce a novel hybrid crow search analysis optimization with an artificial neural network (HCSAOANN). The findings demonstrate that it outperforms in high accuracy, precision, F1-Score, and recall. In the proposed HCSAOANN algorithm, to explore the feature space, we merged the crow search optimization (CSO), which can converge into the overall best solution in the search function, with the upgraded version of the crow search analysis method. Some performance criteria were applied in the studies using an artificial neural network (ANN) as a classifier. The HCSAOANN methodology outperformed as compared to the previous fuzzy technique and achieved 98% accuracy and a 98% precision rate, which is 2.2% better than the previous CSO-ANFIS technique and 8.87% superior to the FC-ANN technique.

Keywords CSA · ANN · CSO · Classifier · Recall · Accuracy

1 Introduction

As the cost of Internet connections and information processing decreases, more and more businesses are exposing themselves to a variety of cyber risks. According to a recent study by the Coordination Centre/Computer Emergency Response Team (CC/CERT) [1], the

✉ Chetan Gupta
chetangupta.gupta1@gmail.com

Amit Kumar
amit.kumar@juet.ac.in

Neelesh Kumar Jain
neesh.jain@juet.ac.in

¹ Computer Science and Engineering JUET, Guna, MP 473226, India

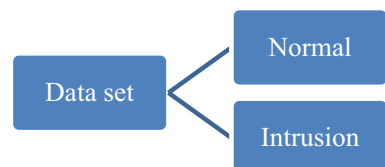
frequency of cyber-attacks has tripled annually. Making our information systems resilient to and tolerant of such attacks has taken on increased significance, particularly for those used for crucial operations in the commercial and defense sectors. Intrusion detection includes detecting malicious actions that put the accessibility, confidentiality, and integrity of data sources in danger. A thorough understanding of recognized attack signatures serves as the cornerstone of traditional intrusion detection methods [2]. Events are compared to signatures to identify intrusions when they are monitored [3]. These methods use features from various audit streams and a collection of malicious activities provided by human experts, comparing attribute values to identify intrusions [4]. Every time a new type of infiltration is identified, the signature database needs to be manually updated. The fact that signature-based approaches are unable to identify evolving cyber threats—which by definition are launched using previously unheard-of tactics—is a significant flaw in their design [5]. Even after a new attack has been found and its signature established, there can occasionally be a significant delay in its dissemination across networks. Due to these restrictions, data mining-based intrusion detection methods are being used more frequently [6].

Techniques for detecting intrusions using data mining are often separated into two classes: The first one is misuse detection, and the second one is anomaly detection. In misuse detection, each occurrence in such a dataset is categorized as either "normal" or "intrusion," as shown in Fig. 1. An algorithm for machine learning is then trained on the indexed data. As long as the attacks have been accurately tagged, these techniques may continuously train up models for intrusion detection using various input data that includes the latest attack forms. Using misuse detection techniques provides a distinct advantage in detecting known attacks and their variants with high accuracy. The fact that systems are not able to identify threats whose occurrences have not yet been observed is clearly a drawback for them. On the other hand, anomaly detection creates typical behavior models and automatically recognizes one departure from all of this, marking the latter as suspicious [7].

Detecting deviations from standard usage patterns is how anomaly detection systems identify new incursions [2]. While having good detection techniques, the possibility of false alarms is a potential drawback of using these methods. Due to its ubiquity and susceptibility. Ensuring security in cyberspace requires ongoing assessment. Massive cyber-attacks have been seen worldwide, including data breaches that gave hackers access to billions of debit and credit card numbers, crypto-jacking attacks, corporate espionage, cyber warfare, social engineering attacks, IoT (Internet of Things) assaults, etc. In addition to the standard system flaws that were exploited in all successful attacks, an IDS, in addition to other security measures such as secure data storage authorization and authentication, methods, and firewalls, is also strongly advised [8].

While a few IDS monitor network traffic and issue alerts for suspicious activity, others also act once threats are discovered, depending on their detection criteria [9]. The meticulous examination of system activity and differentiation between legitimate and malicious activity allows anomaly-based IDS to detect network intrusions. Most IDSs employ misuse detection approaches, which involve scanning and alerting for "known

Fig. 1 Occurrences in dataset



patterns" of harmful behavior. Although this type of IDS restricts itself by consulting the list of known attacks, Their main drawback is that they will be ineffective at protecting against future attacks because signatures have not yet been included. As a result, the system has a significant security hole that a hacker can readily use to trick the IDS. Such IDS must be regularly upgraded to recognize newly emerging attack signatures and those already well-known. Integrating IDS with intelligence often involves leveraging deep learning and machine learning techniques [10]. These methods simplify identifying all types of assaults and protecting the system from all potential threats. But picking the correct dataset is crucial to constructing a machine learning model effective at intrusion detection. On the publicly accessible IDS datasets NSL-KDD [14], KOYOTO [13], KDD 99 [12], and DARPA [11], various machine learning algorithms currently in use are applied. The fact that the statistics, as mentioned earlier, are outdated and don't reflect current traffic patterns is a crucial shortcoming of the planned systems utilizing them. Additionally, numerous researchers have suggested models using machine learning for IDS, and the majority believes accuracy to be the most crucial criterion for assessing the proposed models. But instead, because IDSs make accurate predictions in real-time, accuracy alone is not enough to evaluate a system's performance.

Evaluation of IDS for prediction latency, also known as prediction accuracy, is equally essential to accuracy evaluation. Most academics haven't examined the effect on prediction latency, even though it increases prediction accuracy. A significant actual positive prediction rate should also be included in high-performing IDS, precision, and accuracy latency. False positives, incorrectly identified as attacks, and false negatives, wrongly classified as benign, cannot be considered equal. False negatives can render the system utterly ineffective, whereas positives can add to the system's resources. As a result, in addition to prediction, recall rate, and accuracy, the delay is crucial for assessing the performance of the IDS model.

The remainder of the study is structured as follows: Sect. 3, "Related Work," discusses prior research on machine learning-based intrusion detection. The implementation is covered in detail in Sect. 4, "Proposed Work," along with a thorough explanation of the stages involved in data preparation. The results are discussed in Sect. 5, "Experimental Results." The summary of the developments in Sect. 6, "Conclusion" serves as the paper's conclusion.

1.1 Contribution/Advantage

The main contributions of the proposed work can be summarized as follows:

- (1) The major benefit of the hybrid crowd search analysis and optimization algorithm is that it considers all the features of the NSL-KDD dataset because this dataset contains numerical as well as categorical data, and the processing of categorical data is difficult, so in this approach, categorical data is labeled by a one-hot encoding technique for the consideration of attack detection.
- (2) Another benefit of the proposed work is that feature selection is done on the basis of correlation. For which 0.5 is considered the threshold limit, if the dataset changes or we work on a real-time dataset, we do need to change the correlation-based feature selection, but if we use the feature selection approach that was used in previous work, we need to change the algorithm as per the complex dataset.

- (3) The HCSAOANN algorithm achieved 98% accuracy and a 98% precision rate, which is 2.2% better than the CSO-ANFIS technique and 8.877% superior to the FC-ANN technique.
- (4) The pre-processing step of feature selection chooses the most significant features while improving classification accuracy and lowering the dimensionality of datasets.

2 Rare Classes for Learning

Due to several unique considerations, such as learning from data streams, dealing with imbalanced class distribution, and identifying network connections, traditional data mining approaches are unsuitable for misusing detection-related problems. The problem of an unbalanced class distribution in intrusion detection for networks is apparent because the category of intrusion that is of concern, incursion, is much more minor and consequently more infrequent than the class that represents typical network activities. In such circumstances, where normal conduct may generally represent 98–99% of the total population, a primary classification that classifies everyone in the class label might approach 98–99% accuracy. The accuracy of the category is inadequate in this case as a benchmark evaluation metric. Evaluating the efficacy of a machine learning approach for the minority class is crucial. ROC testing [13] and measures like accuracy and precision have been applied [15]. A confusion matrix, as depicted in Table 1, is frequently utilized to assess the efficiency of such a machine-learning system [16].

From Table 1, ROC, F1-value, as well as precision can be defined as follow:

$$\text{Precision} = \frac{\text{Cuttectly Detected Attacks}}{\text{Correctly Detected Attacks} + \text{False Alarms}}$$

$$\text{Recall} = \frac{\text{Cuttectly Detected Attacks}}{\text{Correctly Detected Attacks} + \text{False Negative}}$$

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

A few historical algorithms for data mining are better suited to deal with intrusions since they are provided based on a succession of events. Labeling all data is necessary for misuse detection algorithms, but classifying connectivity as legal or intrusive takes time and requires considerable human specialists. Developing anomaly-based models becomes quite complicated due to all these problems. Advanced classification algorithms have

Table 1 Common measures for evaluating (intrusions)

Standard matrix (confusion matrix)		Predicted connections labels	
		Normal	Attacks (intrusions)
Actual connection labels	Normal	TN (true negative)	FP (false positive/false alarms)
	Attacks (intrusions)	FN (false negative)	TP (true positive/correctly or appropriately detected attacks)

been developed to learn from unconventional classes. For instance, the two-stage learning method PN rule [9] is centered on determining the norms. The first level aims to find P-rules covering the majority of invasive instances, whereas the second step finds N-rules and removes false alarms produced in the first stage. Through boost generalization ability, CREDOS [17] is a unique method that initially over fits the training data using ripple-down rules before pruning it. It is commonly recognized in data mining that using many classifiers in conjunction can increase prediction accuracy. The Rare-Boost [13] technique aims to include rare class learning techniques [15]. For every one of the four records in Table 1, Rare-Boost modifies the weight of occurrences differently than the regular boosting procedure, which equivalently upgrades the events' importance. By concentrating on a rare class problem, we have also looked into a conventional association-based classification approach. These top item sets are chosen as "meta-features" after each class has undergone the specific item set generation process. Its primary data set is supplemented with these created features, and the resulting data set is subjected to a standard categorization method. The most excellent rules to put the components into the classifiers are chosen using confidence-like metrics in the current association-based classification methods. These methods, however, might only be effective if the data set has a good representation of each type. A number of the remarkable accuracy item sets may also be helpful for the unusual class problems, provided that their precision isn't extremely low. As a result, in addition to accuracy, ROC and F-value are also used to pick the best item sets to be included in the original data set [18].

3 Related Work

Building systems for intrusion detection frequently use machine learning techniques [19]. In this context, the term "classification" describes machine learning methods to differentiate between safe and unsafe behaviors within a network traffic data collection while creating anomaly-based IDS. In this, it was advised to use an effective system for intrusion detection that uses feature selection and an ensemble classifier [20]. On multi-attack classification, they propose the Ensemble technique, which is CFS-BA, in which correlations are used for the selection of features, followed by an ensemble method based on Random Forest, C4.5, as well as Forest PA (forest by penalizing attributes), including an AOP (average of probabilities) algorithm. They claimed that for the NSL KDD dataset as well as the CICIDS 2017 datasets, the classifier offers 99% accuracy. A crucial weakness in the system is that the researcher hasn't taken this model's time efficiency into account. Presented a hybrid intrusion system that uses prioritized improved support vector machines (SVM) as well as KNN (K-Nearest Neighbors) classifiers on three intrusion detection datasets—Koyotto 2006+, NSL-KDD, and KDD Cup99 [21].

This hybrid system for intrusion detection uses the Naive Bayes approach of feature selection for an optimized SVM to find outliers and decrease the dimensionality of the data set. The classification is then performed using a prioritized KNN classifier. Four modules make up this proposed method: DPM (Data Preprocessing Module), FSM (Feature Selection Module), ORM (Outlier Rejection Module), and DMM (Decision-Making Module). This model uses mutual and feature effects recognition to choose pertinent attributes

based on training time, testing time, and accuracy. The author's use of obsolete datasets to assess the effectiveness of the proposed model, which does not adequately reflect current traffic patterns, is the fundamental weakness of the model. Furthermore, compared to the approaches the author explains, machine learning techniques are more efficient and effective. Since most intrusion datasets are skewed, numerous academics have suggested methods to equalize the dataset to improve detection rates. The CIC-IDS2018 dataset's skewed classes were balanced using a model developed using the SMOTE oversampling technique [22].

The samples of crooked styles are enlarged in proportion to the mean sample size. They claim to have used Ad-boost, RF, KNN (K Nearest Neighbor), DT (Decision Tree), and LDA to obtain an accuracy of 99% utilizing this method (Linear Discriminant Analysis). Models for intrusion detection frequently employ techniques like the GA (Genetic Algorithm). Although the article says it has a 99% accuracy rate, time-based metrics are not tested for the approach. The GASVM (Genetic Algorithm Support Vector Machine) classification algorithm was applied to a model [23]. For feature selection, a hybrid method is employed. The GA assigns three priorities to the chosen features. Additional classification is done using these features. Numerous deep-learning techniques are also recommended for developing successful IDS. A dynamic method for anomaly detection was presented, and the neural network was developed using LSTM (long short-term memory) [24]. The most recent CIC-IDS 2018 information was utilized. Although the author's proposed strategy had a 96.2% accuracy rate, the time efficiency of the model was not evaluated. By using the CIC-IDS 2018 dataset, an artificial neural network detection model is proposed. They asserted a 99.97% accuracy rate and an average Receiver Operator Characteristic (ROC) curve area of 0.999 [25].

The model was only meant for botnet identification, despite its excellent accuracy rating of 99.97%. Furthermore, the time efficiency of the model is not evaluated. SCDNN was proposed based on the DNN (Deep Neural Network) as well as SC (Spectral Clustering) algorithms [26]. Sensor network, the NSL-KDD datasets, KDD-Cup99, and the proposed approach were used to assess it. The authors asserted that their method surpasses SVM, BPNN (back propagation neural network), Bayes tree, and RF models in terms of attack detection accuracy. The SCDNN's drawback is that it requires empirical rather than theoretical determination of the clusters' k and parameters and tuning of the DNN layers thresholds and weight parameters. The specified model's time efficiency also isn't measured, and the system is tested with out-of-date datasets. Numerous deep-learning methodologies were compared [27].

The other dataset, CIC-IDS 2018, was used to compare the performance of all of these models, such as convolutional neural networks, recurrent neural networks, DNN, restricted Boltzmann machines, deep belief networks, deep auto encoders, and deep Boltzmann machines. The investigation was run on 5% of the whole dataset. Only accuracy and recall rate were used to evaluate the deep learning models. Missing measurements were F-measure and accuracy rate. The novel model DNN was developed, which consists of two recurrent neural networks, a constrained Boltzmann machine, and a feed-forward module [28].

The dataset CIC-IDS 2018 served as the training ground for the model. Any technique did not correct the extreme skewness of this dataset. A few attack groups within the CIC-IDS dataset had no results for this strategy. NIDS (Network-based Intrusion Detection System) and HIDS (Host-based Intrusion Detection System) both use the newly proposed

distributed deep learning approach for the detection of attacks [29]. The researcher proposed SHIA (Scale-Hybrid-IDS-AlertNet), an intrusion detection system for scaled hybrid attacks. The CIC-IDS 2018 dataset appeared highly skewed, and no strategy was used in the presented methodology to balance it. A few attack categories had unsatisfactory results in this method's findings. Some writers also employ advanced learning algorithms like PSO (Particle Swarm Optimisation) and Extreme Learning to boost the intrusion detection system's effectiveness. Extreme machine learning and clustering approaches are two technologies that Novel Framework proposes for adaptive systems, including interactive network intrusion detection systems [30].

The update manager, decision maker, and clustering manager are the three components that make up its proposed Intrusion Detection System. The update manager receives a repair proposal, and the decision-maker assesses the clustering choices, including the clustering manager organizing the training data. On the now-outdated NSLKDD dataset, the proposed system is tested: FLN (fast learning model)-based PSO. Using the KDD99 dataset, this model's performance is assessed [31]. The time efficiency of the model was not assessed, and the suggested method could not recognize all attack types. An ensemble approach called SVM-KNN-PSO (Support Vector Machine—K Nearest Neighbor—Particle Swarm Optimization) for intrusion detection was proposed [32]. They proposed a model that uses experts in an ensemble-based strategy. There are five binary classifiers for each expert. Every lesson considers the views of the expert. Every classifier in the expert repeats the vote for each observation. The outcomes from numerous experts are combined using weighted majority voting. The serial intrusion detection mechanism and Light GBM-based real-time system for detecting intrusions have been introduced [33]. The proposed model employs two strategies to cut time without sacrificing the precision of assault detection. First, the intrusion detection algorithm employs a light GBM (light gradient boosting machine). Second, to efficiently evaluate traffic data, serial intrusion detection is used. Parallel intrusion detection methods with coordination and communication overheads form the foundation of Swift IDS. Additionally, this model is steady at network speeds up to 1.26 Gbps [34].

Earlier studies haven't assessed how quickly a request can be classified as malicious or benign. Delays within the classification process might severely hamper the system's performance and user experience.

4 Proposed Work

Here, we discuss our planned work, including the original CSA-CSO for intrusion detection.

4.1 The Crow Search Algorithm (CSA)

The CSA is an algorithm that was inspired by nature. This algorithm uses the community's collective intelligence to optimize their performance and achieve their objectives. By

studying the behavior of crows, these algorithms can mimic the communication patterns, decision-making processes, and other social interactions characteristic of this species.

Crows are undoubtedly intelligent creatures that live in a flock and have large brains for their size. They also know where to hide their food, which they can remember and retrieve even months later. Additionally, people are embarrassed when performing the mirror test. They can retain looks, and if an unpleasant one is seen, they can communicate complexly to inform the other crows. Like other social animals, crows occasionally commit robberies by carefully observing where other crows hide their food and then stealing it. A crow goes to a different spot far from where the food is hidden when he thinks another one is following him to trick a thief.

4.2 CSA's Framework

Here present populations with problem dimensions are represented as pd , and the number of crows (solutions) represented as NC . The position of all crow i is represented by a vector $s_i^t = [s_{i1}^t, s_{i2}^t, \dots, s_{ipd}^t]f$, at iteration t or $i = 1, 2, 3, \dots, NC$, where s_i^t is the probable position decision in dimension pd for crow i .

In this case, two things might happen if a crow declares, "i really would like to grab from some other crow j ."

1. The crow that follows it, crow i , is not noticed by crow j ; When crow i locates the food that crow j has stowed, he or she changes its placement as Eq. (1).

$$s_i^{t+1} = s_i^t + ri * fm_j^t * (n_j^t - s_i^t) \quad (1)$$

where ri is a random digit between $[0, 1]$ and fm is the flight length.

2. Once j realizes she is a crow, i follow her to find where her food is being hidden. In this scenario, the crow j dances erratically to deceive the crow i .

The two examples can be mathematically combined, as shown below;

$$s_i^{t+1} = \begin{cases} s_i^t + ri * fm_j^t * (n_j^t - s_i^t), & rj \geq AWP_i^t \\ \text{Random position selected,} & \text{Otherwise} \end{cases} \quad (2)$$

where random numbers are ri and rj and at t iteration, crow j 's awareness probability is AWP_j^t . The value of fm has an impact on crows' capacity to search. High values of fm support global search, whereas low values support local search [35]. Each crow is evaluated while the method is executed using a precisely defined fitness function. The crows then change their places under the fitness value. Every new post is examined for viability. According to Eq. (3), the crows' memories are updated:

$$m_i^{t+1} = s_i^t + 1, \text{ if } f s_i^t + 1 \text{ is better than } f (s_i^t) n_i^t, \text{ Otherwise} \quad (3)$$

Algorithm 1 The CSA

```

1: Crow's location obtain arbitrarily which is initialize as  $s_i^t (1 = 1, 2, 3, \dots, m)$ .
2: fm and AWP have set values.
3: The fitness function calculated  $f (s_i^t)$  for the entire crow  $s_i^t$ .
4: Memory is set as  $n_i^s$ , for all crows  $s_i^t$ ,
(i). set  $t = 0$ 
(ii). Maximum number of iteration =  $x$ 
(iii). Flocks of crows =  $y$ 
5: while  $(t < x)$  do
6:     for  $k = 1$  to  $y$ 
7:         Pick a crow at random to follow.
8:         Give a probability breakdown for awareness
9:         if  $(r \text{ and } AWP_k^t)$ 
10:             $s_k^{t+1} = s_k^t + rk * \text{rand} * (n_k^t - s_k^t)$ 
11:        else
12:             $s_k^{t+1} = \text{locate a position in the search field randomly}$ 
13:        end
14:    end
15:    Analyze the viability of the new location  $s_k^{t+1}$ 
16:     $f(s_i^{t+1})$  evaluates the new position of the crow.
17:    Following that,  $n_i^{t+1}$  updates the crow memory position.
18: end while
19: Produced an optimized solution

```

Local optima are then determined using Eqs. (5) and (7); under those results, the current solution for every crow is updated.

$$S_i^t = [s_{i,1}^t, s_{i,2}^t, \dots, s_{i,d}^t] \quad i = 1, 2, 3 \dots NC \tag{4}$$

$$V_{i,j}^{t+1} = w'v_{i,j}^t + A1r1(G1best_{i,j}^t - s_{i,j}^t) + A2r2(G2best_{i,j}^t - s_{i,j}^t) \tag{5}$$

where A1 and A2 are, respectively, the cognitive and interpersonal factors, the values r1 and r2 are random. The inertia weight, or w', describes how the prior velocity of the swarm influences the velocity of the successive stage. The value of w' is stated in Eq. (6).

$$w' = w'_{\max} - \text{iteration} * \frac{w'_{\max} - w'_{\min}}{x} \tag{6}$$

where; $x = \text{max_iteration}$

$$S_{i,j}^{t+1} = s_{i,1}^t + V_{i,j}^{t+1} \tag{7}$$

4.3 The HCSAOANN (Proposed Algorithm)

Algorithm 2 outlines the key steps of the proposed approach to dealing with feature selection problems. This part also uses the integration of CSA-CSO principles with ANN algorithms to produce an algorithm that could profit from its combination.

The efficiency of fully random following every crow in the classic CSA is increased in the proposed technique by targeting a few selected crows with the greatest goodies. Then, algorithm 3 uses the CSA approach to build the various crow positions or change places in the CSO. The goal is to allow the two algorithms to explore the search space independently without being influenced by the results of either strategy. Establish the initial values for the crow search parameters, including awareness likelihood, flock size, flight duration, crow position, and the number of iterations. Create an attack prediction including a classification model using a neural network, and then optimize it using a crow search-based approach with each update of the crow memory. The selecting properties of the NSL-KDD dataset were subjected to the Met heuristic Crow Search Optimization (CSO) method in contrast to the mentioned state-of-the-art classifier, which improved the accuracy and detection rate of the IDS while decreasing the false alarm rate.

Algorithm 2 The HCSAOANN

```

1: Establish the crow's size N' with dimension D' having the same attributes as
the number dataset.
2: Get the A1, A2; Maximum velocity  $v'_{max}$ ; weight factor  $w'_{min}, w'_{max}$ ; Awareness
probability AWP; Maximum iteration x; flight length fl'
3: For each solution, the population is determined at random by  $s_i^t$  and setting
the D-dimensional vector's velocity to  $t = 0$ . The D dimensional vector as the
velocity set  $t = 0$ , population gets randomly set as for each solution,
4:  $w' = w'_{max} - \text{iteration} * \frac{w'_{max} - w'_{min}}{x}$ 
5: Whenever G1best and G2best are set, the optimal solutions for all of the
solutions .are evaluated with  $f(s_{i,j}^t)$ 
6: Initialization of parameters while using ANN, CSA, and CSO.
7: Run Apply Neural Network model to develop a model for the classification and
attack prediction.
8: At every iteration; the crow's memory updating, improve this model by using
Crow Search-based optimization.
9: The NSL-KDD dataset's selective features were subjected to the Metaheuristic
Crow Search Optimization (CSO) algorithm.
10: Run CSA through  $s_{i,j}^t$  as the population, a group of crows with that must eat
the best food, as well as a minimum number of crows.
11: Utilizing algorithm 3, reversely modify the CSA's return position..
12: Update the crows' location
13:   for l = 1 to XX
14:      $v_i^{t+1} = w'v_i^t + A1ni,1(G1best_i^t - s_i^t) + A2ni,2(G2best_i^t - s_i^t)$ 
15:   end
16:   for l = 1 to XX
17:     for j = 1 to D
18:   if ( $v'(i,j) > V'_{max}$ )
19:  $v'(i,j) = V'_{max}$ 
20:   end
21:   if ( $v'(i,j) < -V'_{max}$ )
22:  $v'(i,j) = -V'_{max}$ 
23:   end
24:  $u = \frac{1}{1 + e^{-v'(i,j)}}$ 
25:   if (rand < u)
26:      $s_{i,j}^{t+1} = 1$ 
27:   else
28:      $s_{i,j}^{t+1} = 0$ 
29:   end
30:   end
31:   end
32:  $t' = t' + 1$ 
33: Get best solution

```

Algorithm 3 Crow location in opposite

```

1: do
2:   i = 1
3:   i = i++
4:   if ((f(sit) > mincrowt))
5:     sit/OPP = ai + bi - sit
6:   while (i <= N)

```

When the feature vector size is N , the number of potentials picked tends to be 2, which is a fairly large space for comprehensive searching. By accomplishing this and providing the ideal feature combination, the recommended hybrid algorithm is employed to investigate the feature space dynamically. Here, feature selection must satisfy many objectives to achieve the optimum result, which minimizes the feature subset selected and simultaneously increases the output accuracy for a given classifier. The flow chart of the proposed methodology is illustrated in Fig. 2.

1. Data pre-processing:

- a. Read the input NSL-KDD dataset.
- b. Remove the attribute 'difficulty_level' from the dataset since we are assuming it is not a decision parameter and it will not help us to classify our dataset.
- c. Changing the attack labels to their respective attack classes.
- d. Selecting numeric attribute columns from the dataset.

2. Data normalization:

- a. Normalize the dataset using a standard scaler, since normalization can help train of our neural networks as the different features are on a similar scale, which helps to stabilize the gradient descent step, allowing us to use larger learning rates or it will help models converge faster for a given learning rate.
- b. Selecting categorical data attribute from the dataset.
- c. Creating a data frame from this categorical attributes dataset.
- d. Apply the one-hot-encoding attack label for categorical attributes using pandas. get_dummies() function to categorize these categorical column attributes and used them in the learning model.

3. Feature selection:

- a. Finding the attributes that have more than 0.5 correlations with the encoded attack label attribute when considering only the dependent attributes from the dataset.
- b. Selecting attributes is done by using Pearson correlation coefficient.
- c. Joining the selected attribute with the one-hot-encoded categorical data frame.

4. Training and testing:

- a. Splitting the dataset into 75% for training and 25% for testing or to get the more accurate model we can split it into more training and testing sets.

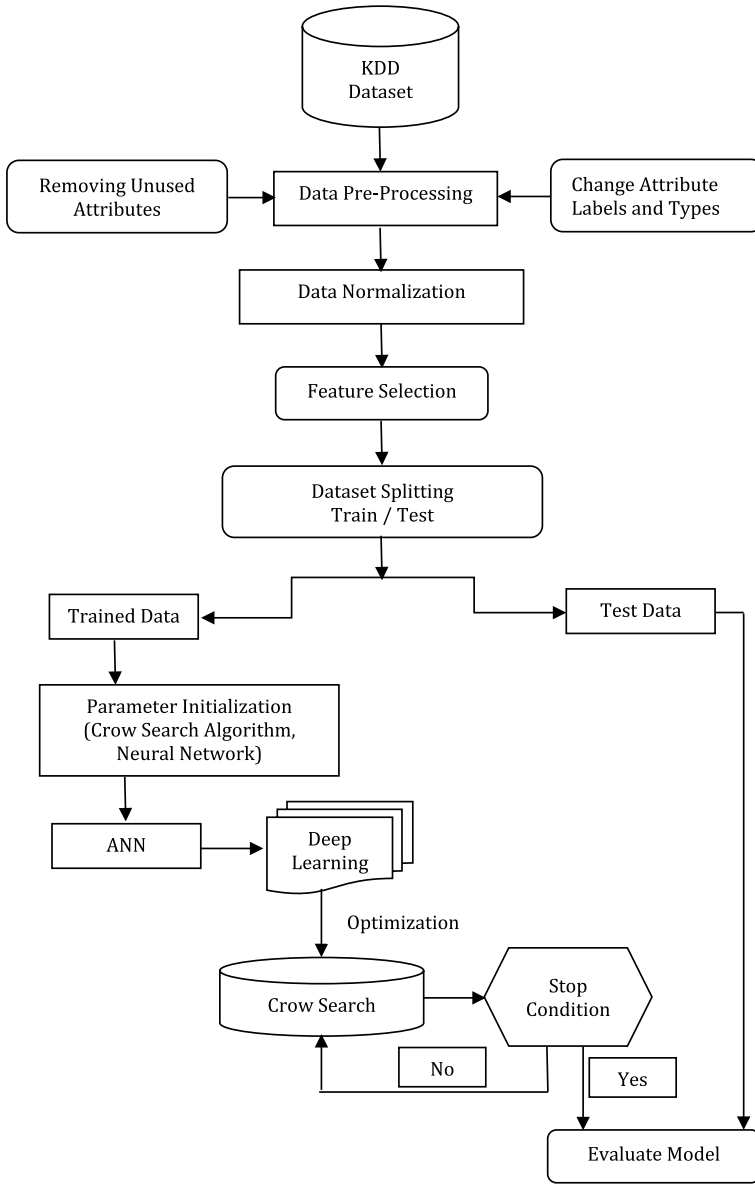


Fig. 2 HCSAOANN proposed work

5. Parameter initializing and model evaluation:

- a. Initialize the crow search parameters, such as awareness probability, flight length, flock size, maximum number of iterations, and position of the crow.
- b. Apply the neural network model to create a model for attack prediction and classification. Optimize this model at each iteration of the Crow memory update by applying Crow search-based optimization. The meta-heuristic Crow Search Opti-

mization (CSO) algorithm was applied to the selective features of the NSL-KDD dataset, which cut down the false alarm rate and enhanced the detection rate and accuracy of the IDS as compared with the mentioned state-of-the-art classifiers.

5 Experimental Results

The observations in this research are conducted over a few stages:

1. Analysis of various attacks in a given dataset (NSL-KDD Dataset)
 - a. Classification Attacks in the NSL-KDD Dataset.
 - b. Classification of Major Attacks in the given dataset (NSL-KDD Dataset).
2. Various plots of accuracy vs. epoch for training and test datasets based on the number of neurons, epochs, and batch size
 - a. Show some results on when we applied with CSA and without CSA.
 - b. Performance Analysis of Various Metrics on Binary Class Distribution.
 - c. Performance Analysis of Various Metrics on Multi-Class Distribution.

5.1 Definition of Dataset

The improved version of KDD99, known as NSL-KDD (National Security Lab-Knowledge Discovery and Data Mining), was developed to overcome its drawbacks. Downloading this publicly accessible dataset is possible at <https://www.kaggle.com/datasets/hassan06/nslkdd>. The train and test sets' duplicate records are first removed. Second, alternative records from the original KDD99 dataset were selected to produce trustworthy results from classifier systems. Third, the imbalanced probability distribution problem was solved. The NSL-KDD data collection consists of 125,973 training examples and 22,544 test cases.

5.2 Result Discussion

The algorithms employed in this work were initiated at random, permitted to execute $P = 10 +$ individually with $60 +$ iterations, and kept at a population size of 20, with a dimension equal to the figure of characteristics in the given dataset. The specifications used for the comparative methods are listed in Table 3. All of the tests for this study were done in Python on a machine with an Intel (R) Core (TM) i3-7020U CPU at 2.30 GHz. The following statistical metrics are noted based on the validation results.

Applying Crow Search Optimization, in this case, enhances the algorithm's predictive power. NSL-KDD data sets are publicly available data sets for assessing intrusion detection systems. Some of the underlying issues are resolved by the NSL-KDD data set.

The set of numerous duplicated records in the NSL-KDD data set, which biases learning algorithms towards frequent records and prevents them from learning uncommon records that are typically more destructive to networks, like U2R and R2L attacks, is one of the data set's most significant efficiencies. In this experiment, we used the NSL-KDD data set, which contains 494,021 records, including 125,973 records.

Figure 3 shows that the data set is severely out of balance or that there is not a consistent distribution of samples for every type of attack. DoS-type attacks have a large sample

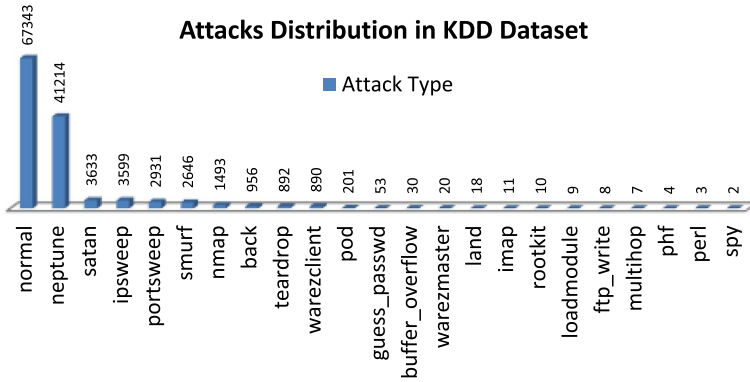


Fig. 3 Attacks distribution in KDD dataset

count, whereas U2R-type attacks have a lower sample count. This uneven distribution of samples could be problematic for any classification model based on data mining.

A variety of fundamental attacks from the KDD Cup dataset are shown in Table 2. The different attack types from the dataset are divided into Normal, Probe, Dos, U2R, and R2L groups.

The KDDCup dataset’s analysis of all potential assaults is shown in Fig. 4. Without using any distributions of attack types, the analysis is complete. This data collection includes one normal type and four different forms of attack. Using the given NSL-KDD data set, an experiment is conducted to evaluate the robustness of the model.

Probe: When an attacker uses to host and network scanning techniques to access data on the target network. A port scanning is one example of a probe.

DoS: Whenever an attacker prevents authorized users from accessing a system or service.

U2R: Whenever a hacker tries to convert a restricted user’s advantages to root access (During stolen or malware facts infection).

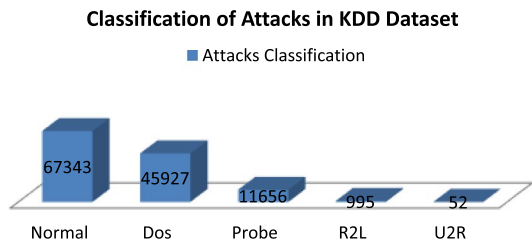
R2L: An outsider gains wireless connectivity to a targeted system by imitating other nearby users. As they imitate the behavior of typical users, U2R, as well as R2L attacks, were some of the most difficult to recognize.

Figure 5 shows the attack type distribution from the KDDCup dataset. The classification between normal and abnormal labels is shown in Fig. 5a. Attacks of the normal and

Table 2 Classification attacks in the NSL-KDD dataset

	Total	Normal	Dos	Probe	R2L	U2R
KDD Dataset	125,973	67,343	45,927	11,656	995	52

Fig. 4 Classification of attacks in KDD dataset



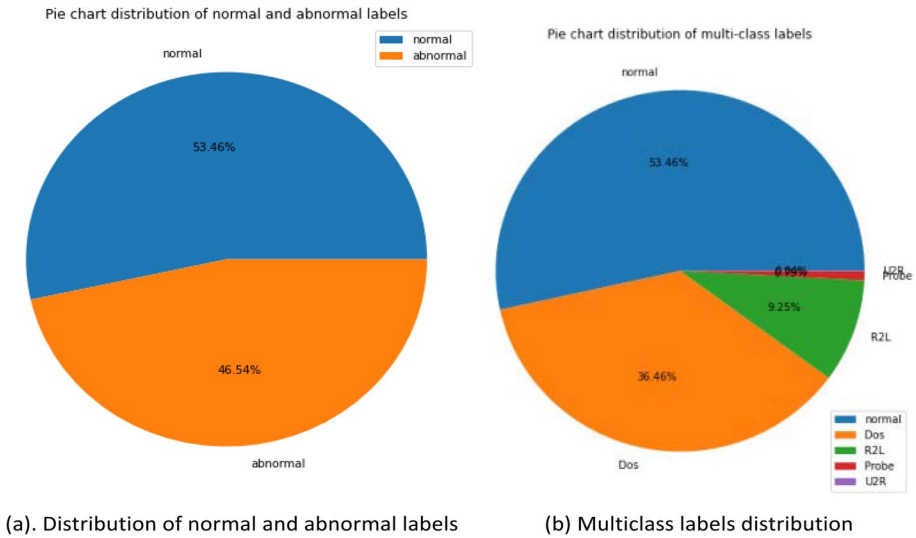


Fig. 5 Classification of major attacks in the NSL-KDD dataset

atypical types account for 53.46 and 46.54%, respectively. Conversely, Fig. 5b shows how multiclass labels are classified, with normal remaining at 53.46%, DoS at 36.46%, R2L at 9.25%, and Probe and U2R at 84%.

Five possible classifications were covered in training and testing sessions: normal, probe, dos, U2R, and R2L. Here, the normal class denotes the absence of anomalies. As the last classes influence the results, these five classes are considered. These categorizations remain useful for identifying intrusions and remain fairly unbalanced (a crucial aspect of intrusion data), having sufficient cases for each class to produce more useful results.

Table 3 shows the dataset's most dependent quality ordering distribution, along with a list of attributes and their correlation coefficients. The correlation coefficient, which identifies the dataset's primary indicators of attack types, serves as the foundation for the sorting.

The accuracy of the proposed algorithm changes according to the testing and training of KDDCup datasets, as illustrated in Fig. 6. The accuracy is calculated using the split between the training sets, test sets, and the number of epochs. 50 neurons, 10 epochs,

Table 3 Ordering distribution of most dependent attributes

Attribute	Correlation coefficient
Count	0.576444
srv_serror_rate	0.648289
serror_rate	0.650652
dst_host_serror_rate	0.651842
dst_host_srv_serror_rate	0.654985
logged_in	0.690171
dst_host_same_srv_rate	0.693803
dst_host_srv_count	0.722535
same_srv_rate	0.751913
Intrusion	1

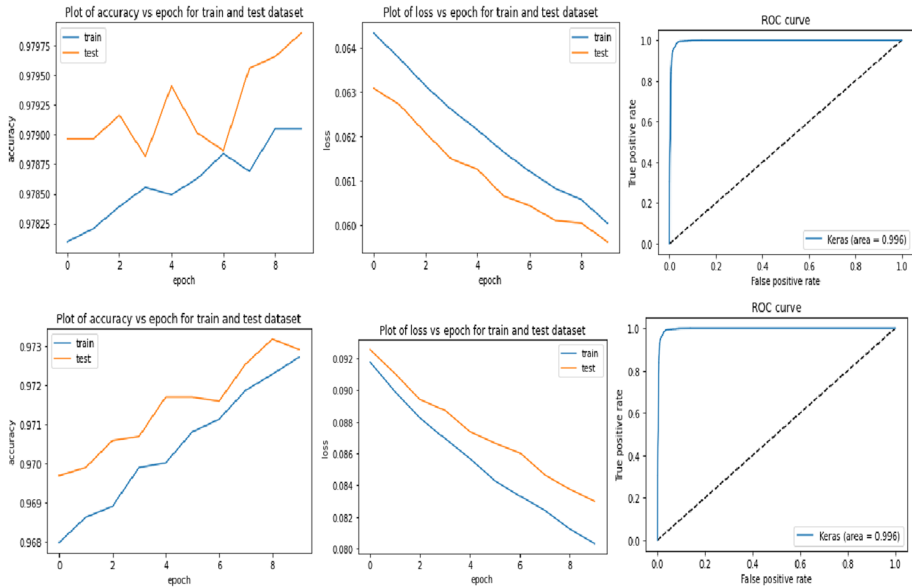


Fig. 6 Plot of accuracy versus epoch for train and test dataset based on 50 neurons, 10 epochs and 500 batch size

and 500 batch sizes were used for the analysis, as illustrated in Fig. 6. The analysis uses various neurons and 500 batches for 500 epochs, as indicated in Table 4 and Table 5 for binary and multi-classification, respectively.

We are using here four parameters like average loss, accuracy, F1 score, precision, and recall at each epoch, and we analyzed these parameter values along with CSA and without CSA. After the analysis at each epoch, we found average loss to be very less with CSA than without CSA; on the other hand, accuracy, recall, and F1 score values were found to be very high with CSA compared to without CSA.

Figure 7 illustrates the study and comparison of our works, correctness for various epochs by categorizing the dataset's binary distribution. The graphs illustrate the effectiveness of detecting intrusions using a binary set created using crow search optimization for 50 neuron layers in neural network-based models.

Figure 8 illustrates the study and comparison of our works, correctness for various epochs by categorizing the dataset's binary distribution. The graphs illustrate the effectiveness of neural network-based models utilizing crow search optimization for 100 neuron layers in identifying intrusions as a binary set.

Figure 9 compares and analyses the accuracy of our work by categorizing the dataset's binary distribution across multiple epochs. The graphs illustrate the effectiveness of detecting intrusions using a binary set generated using Crow Search Optimization for 500 neuronal layers.

Figure 10 represents the analysis and comparison of our proposed work in terms of precision for various epochs by classifying the binary distribution of the dataset. The graphs show the performance of detecting intrusions as a binary set by an optimization model based on a neural network using Crow Search Optimization for 50 neural layers.

Table 4 Performance analysis of various metrics on binary class distribution

Binary class distribution				NN with CSA				NN without CSA				
No. of neurons	No. of epochs	batch size	No. of neurons	No. of epochs	batch size	No. of neurons	No. of epochs	batch size	No. of neurons	No. of epochs	batch size	No. of neurons
50	10	500	50	10	500	50	10	500	50	10	500	50
50	50	500	50	50	500	50	50	500	50	50	500	50
50	100	500	50	100	500	50	100	500	50	100	500	50
50	200	500	50	200	500	50	200	500	50	200	500	50
50	500	500	50	500	500	50	500	500	50	500	500	50
100	10	500	100	10	500	100	10	500	100	10	500	100
100	50	500	100	50	500	100	50	500	100	50	500	100
100	100	500	100	100	500	100	100	500	100	100	500	100
100	200	500	100	200	500	100	200	500	100	200	500	100
100	500	500	100	500	500	100	500	500	100	500	500	100
500	10	500	500	10	500	500	10	500	500	10	500	500
500	50	500	500	50	500	500	50	500	500	50	500	500
500	100	500	500	100	500	500	100	500	500	100	500	500
500	200	500	500	200	500	500	200	500	500	200	500	500
500	500	500	500	500	500	500	500	500	500	500	500	500

Table 5 Performance analysis of various metrics on multiclass distribution

Multi class distribution				NN with CSA						NN without CSA								
No. of neurons	No. of epochs	batch size		Average loss	Accuracy	Precision	Recall	F1 score	Average loss	Accuracy	Precision	Recall	F1 score	Average loss	Accuracy	Precision	Recall	F1 score
50	10	500		0.08	0.97	0.97	0.97	0.97	0.57	0.92	0.9	0.91	0.91	0.57	0.92	0.9	0.91	0.91
50	50	500		0.06	0.98	0.98	0.98	0.98	0.19	0.94	0.92	0.9	0.9	0.19	0.94	0.92	0.9	0.9
50	100	500		0.06	0.98	0.98	0.98	0.98	0.37	0.92	0.94	0.92	0.92	0.37	0.92	0.94	0.92	0.92
50	200	500		0.06	0.98	0.98	0.98	0.98	0.12	0.91	0.9	0.92	0.93	0.12	0.91	0.9	0.92	0.93
50	500	500		0.06	0.98	0.98	0.98	0.98	0.37	0.91	0.93	0.9	0.9	0.37	0.91	0.93	0.9	0.9
100	10	500		0.08	0.97	0.98	0.97	0.97	0.46	0.93	0.91	0.92	0.91	0.46	0.93	0.91	0.92	0.91
100	50	500		0.06	0.98	0.98	0.98	0.98	0.51	0.9	0.94	0.92	0.9	0.51	0.9	0.94	0.92	0.9
100	100	500		0.06	0.98	0.98	0.98	0.98	0.32	0.91	0.93	0.94	0.93	0.32	0.91	0.93	0.94	0.93
100	200	500		0.06	0.98	0.98	0.98	0.98	0.38	0.91	0.91	0.93	0.91	0.38	0.91	0.91	0.93	0.91
100	500	500		0.07	0.98	0.98	0.98	0.98	0.14	0.94	0.94	0.92	0.94	0.14	0.94	0.94	0.92	0.94
500	10	500		0.07	0.98	0.98	0.98	0.98	0.36	0.93	0.91	0.92	0.93	0.36	0.93	0.91	0.92	0.93
500	50	500		0.06	0.98	0.98	0.98	0.98	0.48	0.94	0.9	0.94	0.92	0.48	0.94	0.9	0.94	0.92
500	100	500		0.06	0.98	0.98	0.98	0.98	0.14	0.93	0.93	0.92	0.91	0.14	0.93	0.93	0.92	0.91
500	200	500		0.06	0.98	0.98	0.98	0.98	0.33	0.93	0.91	0.93	0.92	0.33	0.93	0.91	0.93	0.92
500	500	500		0.07	0.98	0.98	0.98	0.98	0.59	0.91	0.92	0.9	0.93	0.59	0.91	0.92	0.9	0.93

Fig. 7 Comparison of accuracy on 50 neurons for binary distribution

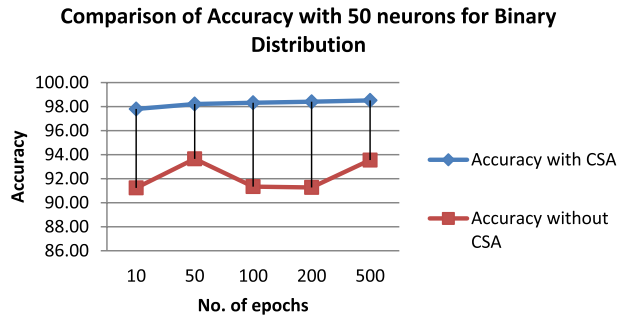


Fig. 8 Comparison of accuracy on 100 neurons for binary distribution

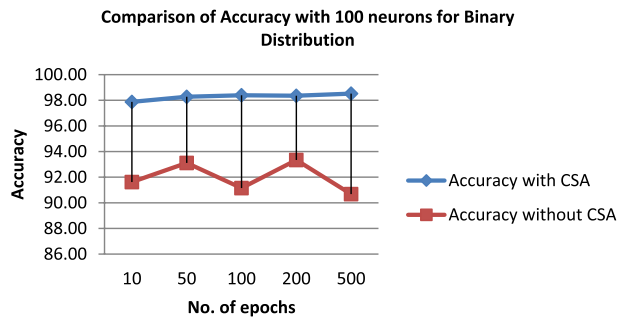


Fig. 9 Comparison of accuracy on 500 neurons for binary distribution

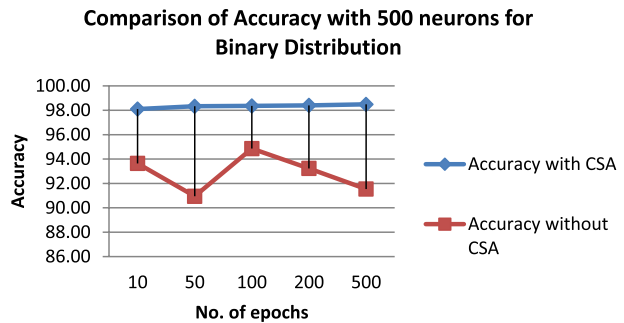


Fig. 10 Comparison of precision on 50 neurons for binary distribution

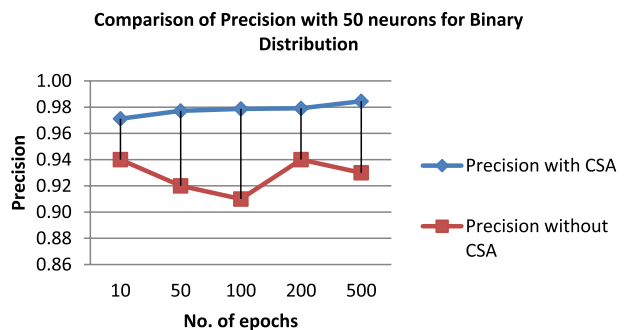
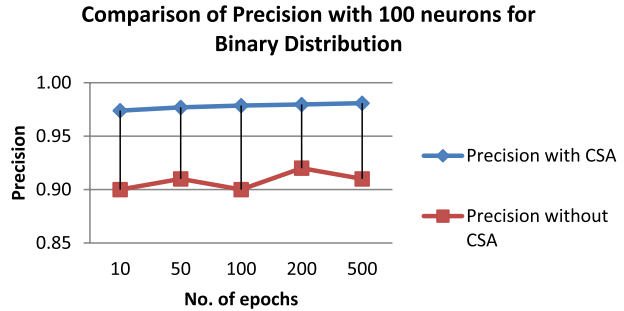


Fig. 11 Comparison of precision on 100 neurons for binary distribution



The dataset’s binary distribution is classified in Fig. 11 to show the study and comparison of our proposed work Precision for different epochs. The graphs demonstrate the effectiveness of detecting intrusions using a binary set created by optimizing neural network-based models using Crow Search Optimization for 100 neuron layers.

Figure 12 represents the analysis and comparison of our proposed work in terms of Precision for various epochs by classifying the binary distribution of the dataset. The graphs show the performance detecting intrusions as a binary set by optimization model, based on Neural Network using Crow Search Optimization for 500 neurons layers.

Figure 13 compares and analyses our proposed work accuracy for various epochs by classifying the dataset’s multiclass distribution. The graphs illustrate the effectiveness of identifying intrusions using a multiclass set of optimizations for 500 neuronal layers of neural network-based models.

The multiclass distribution of the given dataset and the analysis and comparison of our work in Precision for different epochs are shown in Fig. 14. The graphs illustrate the

Fig. 12 Comparison of precision on 500 neurons for binary distribution

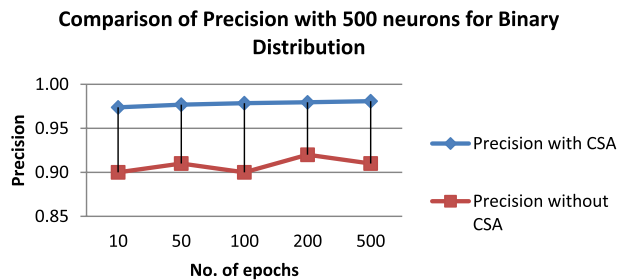


Fig. 13 Comparison of precision on 500 neurons for multiclass distribution

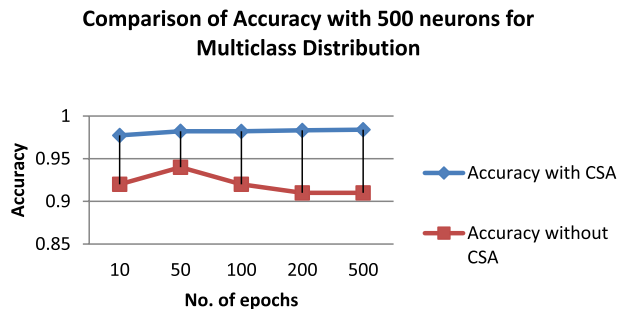
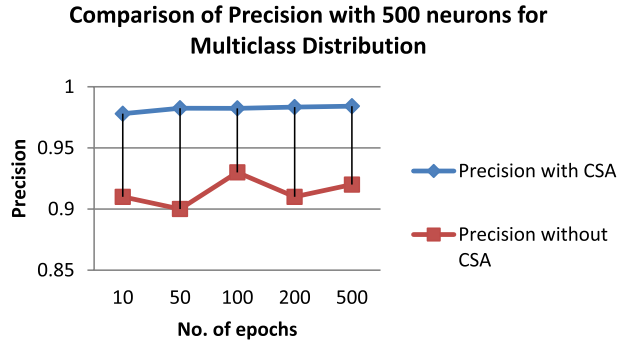


Fig. 14 Comparison of accuracy on 500 neurons for multiclass distribution



effectiveness of identifying intrusions using a multiclass set of optimizations for 500 neuronal layers of neural network-based models.

By classifying the dataset’s multiclass distribution, the analysis and comparison of our proposed work regarding a recall for different epochs are shown in Fig. 15. The graphs illustrate the effectiveness of identifying intrusions using a multiclass set of optimizations for 500 neuronal layers of neural network-based models.

Table 4 shows the performance analysis of various metrics on binary class distribution for different sizes of neurons using both neural networks with crow search analysis and neural networks without crow search analysis.

The performance analysis of various metrics on multiclass distribution for neurons of various sizes using both neural networks with crow search analysis and neural networks without crow search analysis is shown in Table 5.

The result of the intrusion detection based on the NSL-KDD dataset is better and more efficient as compared with the previous approach because our detection rate is 98%, as shown in Table 6, which is 2.2% better than the CSO-ANFIS technique as shown in Fig. 16. And the precision, recall, and f1 score also reached up to 98% accuracy, which shows the effectiveness of this proposed approach.

Fig. 15 Comparison of recall on 500 neurons for multiclass distribution

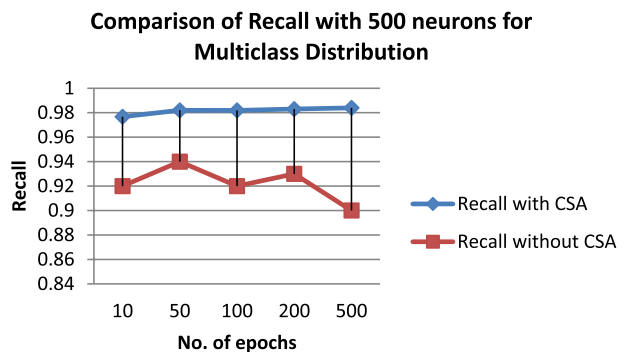
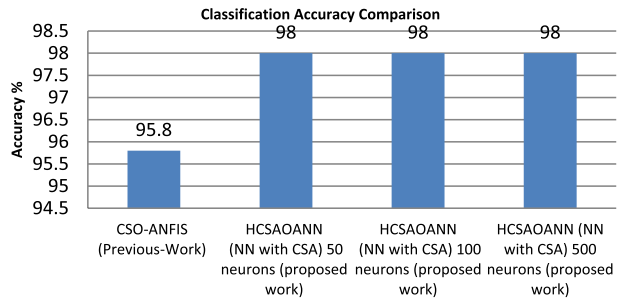


Table 6 Classification accuracy comparison

Method	Accuracy
CSO-ANFIS (Previous-work) [1]	95.80%
HCSAOANN (NN with CSA) 50 neurons (proposed work)	98%
HCSAOANN (NN with CSA) 100 neurons (proposed work)	98%
HCSAOANN (NN with CSA) 500 neurons (proposed work)	98%

Fig. 16 Graphical plot of accuracy, based on 50/100/500 neurons for multiclass distribution

6 Conclusion

In this paper, we propose a hybridized method combining a neural network-enhanced crow search algorithm with a crow search optimization strategy to help remove features from irrelevant and noisy datasets, otherwise recognized as a feature selection strategy. The recommended algorithm is the Hybrid Crow Search Analysis and Optimization Algorithm with Neural Network (HCSAOANN). This combined algorithm performs better than any one of its component algorithms separately. The proposed algorithm, HCSAOANN, was validated and studied over NSL KDD repository datasets to ensure the algorithm's validity and accuracy. Based on some factors, the HCSAOANN algorithm was compared to many well-known optimization techniques (CSA-CSO and ANN). These factors include the number of characteristics chosen and the accuracy of the classifier's outcome. The comparison tests and analyses demonstrate that the recommended HCSAOANN performs the best on the dataset in terms of overall classification accuracy. The proposed model achieved 98% accuracy and a 98% precision rate, which is better than the previous technique.

In conclusion, the outcomes demonstrate that the HCSAOANN can compete with other algorithms and achieve superior outcomes. There are major practical ramifications for the proposed HCSAOANN feature selection mechanism. The HCSAOANN can be useful to different industries, including data science, engineering, sentiment analysis, etc. By improving this method, a new selection technique might be used to select the least likely traits that could potentially be the subject of future research. Real issues like sentiment analysis, including cancer detection, may be looked into in the future using HCSAOANN. Various dataset types can be used to test HCSAOANN.

Funding No funding was received for conducting this study.

Data Availability The data and material of the manuscript is available.

Code Availability The code is available in GitHub.

Declarations

Conflict of interest There is no conflict of interest.

References

1. Manimurugan, S., Majidi, A.-Q., Mustaffa Mohammed, C., & Narmatha, R. V. (2020). Intrusion detection in networks using crow search optimization algorithm with adaptive neuro-fuzzy inference system. *Microprocessors and Microsystems*. <https://doi.org/10.1016/j.micpro.2020.103261>
2. Verma, A., & Ranga, V. (2020). Machine learning based intrusion detection systems for IoT applications. *Wireless Personal Communications*, *111*, 2287–2310. <https://doi.org/10.1007/s11277-019-06986-8>
3. Lifandali, O., Abghour, N., & Chiba, Z. (2023). Feature selection using a combination of ant colony optimization and random forest algorithms applied to isolation forest based intrusion detection system. *Procedia Computer Science*, *220*, 796–805. <https://doi.org/10.1016/j.procs.2023.03.106>
4. Almasoudy, F. H., Al-Yaseen, W. L., & Idrees, A. K. (2020). Differential evolution wrapper feature selection for intrusion detection system. *Procedia Computer Science*, *167*, 1230–1239. <https://doi.org/10.1016/j.procs.2020.03.438>
5. Alsarhan, A., Alauthman, M., Alshdaifat, E., Al-Ghuwairi, A. R., & Al-Dubai, A. (2021). Machine Learning-driven optimization for SVM-based intrusion detection system in vehicular ad hoc networks. *Journal of Ambient Intelligence and Humanized Computing*. <https://doi.org/10.1007/s12652-021-02963-x>
6. Dwivedi, S., Vardhan, M., & Tripathi, S. (2021). Building an efficient intrusion detection system using grasshopper optimization algorithm for anomaly detection. *Cluster Computing*, *24*(3), 1881–1900. <https://doi.org/10.1007/s10586-020-03229-5>
7. Singhal, A., Maan, A., Chaudhary, D., & Vishwakarma, D. (2021). A Hybrid machine learning and data mining based approach to network intrusion detection. International conference on artificial intelligence and smart systems (ICAIS), Coimbatore, India, pp 312–318. Doi: <https://doi.org/10.1109/ICAIS50930.2021.9395918>.
8. Sivagaminathan, V., Sharma, M., & Henge, S. K. (2023). Intrusion detection systems for wireless sensor networks using computational intelligence techniques. *Cyber security*. <https://doi.org/10.1186/s42400-023-00161-0>
9. Prasad, S. N., Selvan, K. S., & Dhevi, B. L. (2021). Intrusion detection system in wireless sensor networks and fair resource allocation using geometric deep learning techniques. *Wireless Personal Communications*, *123*(4), 3401–3412. <https://doi.org/10.1007/s11277-021-09294-2>
10. Ka Mushtaq, E., Zameer, A., & Khan, A. (2022). A two-stage stacked ensemble intrusion detection system using five base classifiers and MLP with optimal feature selection. *Microprocessor Microsystem*. <https://doi.org/10.1016/j.micpro.2022.104660>
11. Wu, Z., Wang, J., Hu, L., Zhang, Z., & Wu, H. (2020). A network intrusion detection method based on semantic re-encoding and deep learning. *Journal of Network and Computer Applications*. <https://doi.org/10.1016/J.JNCA.2020.102688>
12. Siddique, K., Akhtar, Z., Khan, F. A., & Kim, Y. (2019). KDD cup 99 data sets: A perspective on the role of data sets in network intrusion detection research. *IEEE Computer*, *52*(2), 41–51. <https://doi.org/10.1109/mc.2018.2888764>
13. Sadhasivan, D. K., & Balasubramanian, K. (2017). A fusion of multiagent functionalities for effective intrusion detection system. *Security Communication Networks*. <https://doi.org/10.1155/2017/6216078>
14. Ingre, B., & Yadav, A. (2015). Performance analysis of NSL-KDD dataset using ANN. *International Conference on Signal Processing*. <https://doi.org/10.1109/spaces.2015.7058223>
15. Ghazy, R. A., El-Rabaie, E. M., Dessouky, M. I., El-Fishawy, N., & El-Samie, F. E. A. (2020). Feature selection ranking and subset-based techniques with different classifiers for intrusion detection. *Wireless Personal Communications*, *111*(1), 375–393. <https://doi.org/10.1007/s11277-019-06864-3>

16. Mohd, N., Singh, A., & Bhadauria, H. S. (2021). Intrusion detection system based on hybrid hierarchical classifiers. *Wireless Personal Communications*, 121(1), 659–686. <https://doi.org/10.1007/s11277-021-08655-1>
17. Gao, B., Bu, B., Zhang, W., & Li, X. (2021). An intrusion detection method based on machine learning and state observer for train-ground communication systems. *IEEE Transactions on Intelligent Transportation Systems*. <https://doi.org/10.1109/TITS.2021.3058553>
18. Sinha, S., & Paul, A. (2020). Neuro-fuzzy based intrusion detection system for wireless sensor network. *Wireless Personal Communications*, 114, 835–851. <https://doi.org/10.1007/s11277-020-07395-y>
19. Ridwan, M. A., Radzi, N. A. M., Abdullah, F., & Jalil, Y. E. (2021). applications of machine learning in networking: A survey of current issues and future challenges. *IEEE Access*, 9, 52523–52556. <https://doi.org/10.1109/ACCESS.2021.3069210>
20. Zhou, Y., Cheng, G., Jiang, S., & Dai, M. (2020). Building an efficient intrusion detection system based on feature selection and ensemble classifier. *Computer Network*. <https://doi.org/10.1016/j.comnet.2020.107247>
21. Saleh, A. I., Talaat, F. M., & Labib, L. M. (2017). A Hybrid intrusion detection system (HIDS) based on prioritized k-nearest neighbors and optimized SVM classifiers. *Artificial Intelligence Review*, 51, 403–443. <https://doi.org/10.1007/s10462-017-9567-1>
22. Karatas, G., Demir, O., & Sahingoz, O. K. (2020). Increasing the performance of machine learning-based IDSs on an imbalanced and up-to-date dataset. *IEEE Access*, 8, 32150–32162. <https://doi.org/10.1109/access.2020.2973219>
23. Aslahi-Shahri, B., Rahmani, R., Chizari, M., Maralani, A., Eslami, M., & Golkar, M. (2015). A hybrid method consisting of GA and SVM for intrusion detection system. *Neural Computing and Application*, 27(6), 1669–1676. <https://doi.org/10.1007/s00521-015-1964-2>
24. Lin, P., Ye, K., & Xu, C.-Z. (2019). dynamic network anomaly detection system by using deep learning techniques. *Cloud Computing Cloud*. https://doi.org/10.1007/978-3-030-23502-4_12
25. Kanimozhi, V., & Prem, J. T. (2019). Artificial intelligence based network intrusion detection with hyper-parameter optimization tuning on the realistic cyber dataset CSE-CIC-IDS2018 using cloud computing. *ICT Express*, 5(3), 211–214. <https://doi.org/10.1016/j.ict.2019.03.003>
26. Ma, T., Wang, F., Cheng, J., Yu, Y., & Chen, X. (2016). A hybrid spectral clustering and deep neural network ensemble algorithm for intrusion detection in sensor networks. *Topology Control in Emerging Sensor Networks*. <https://doi.org/10.3390/S16101701>
27. Ferrag, M. A., Maglaras, L. A., Moschoyiannis, S., & Janicke, H. (2020). Deep learning for cyber security intrusion detection: approaches, datasets, and comparative study. *Journal of Information Security and Applications*. <https://doi.org/10.1016/J.JISA.2019.102419>
28. Atefinia, R., & Ahmadi, M. (2021). network intrusion detection using multi-architectural modular deep neural network. *Journal of Supercomputing*, 77, 3571–3593. <https://doi.org/10.1007/s11227-020-03410-y>
29. Vinayakumar, R., Alazab, M., Soman, K. P., Poornachandran, P., Al-Nemrat, A., & Venkatraman, S. (2019). Deep learning approach for intelligent intrusion detection system. *IEEE Access*, 7, 41525–41550. <https://doi.org/10.1109/ACCESS.2019.2895334>
30. Roshan, S., Roshan, S., Miche, Y., Akusok, A., & Lendasse, A. (2018). Adaptive and online network intrusion detection system using clustering and extreme learning machines. *Journal of the Franklin Institute*. <https://doi.org/10.1016/J.JFRANKLIN.2017.06.006>
31. Ali, M. H., Al Mohammed, B. A. D., Ismail, A., & Zolkipli, M. F. (2018). A new intrusion detection system based on fast learning network and particle Swarm optimization. *IEEE Access*. <https://doi.org/10.1109/ACCESS.2018.2820092>
32. Aburomman, A. A., & IbneReaz, M. B. (2016). A novel SVM-kNN-PSO ensemble method for intrusion detection system. *Applied Soft Computing*. <https://doi.org/10.1016/J.ASOC.2015.10.011>
33. Dongzi, J., Yiqin, L., Jiancheng, Q., Zhe, C., & Zhongshu, M. (2020). SwiftIDS: Real-time intrusion detection system based on lightGBM and parallel intrusion detection mechanism. *Computers & Security*. <https://doi.org/10.1016/J.COSE.2020.101984>
34. Yu, Y., & Bian, N. (2020). An intrusion detection method using few-shot learning. *IEEE Access*, 8, 49730–49740. <https://doi.org/10.1109/ACCESS.2020.2980136>
35. Chouhan, N., Khan, A., & Khan, H.-U.-R. (2019). Network anomaly detection using channel boosted and residual learning based deep convolutional neural network. *Applied Soft Computing*. <https://doi.org/10.1016/j.asoc.2019.105612>

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.



Chetan Gupta did M. Tech. from Technocrats Institute of Technology, Bhopal in Computer Science and Engineering with (with Honors) in 2014 and did B.E. from All Saints' College of Technology, Bhopal in Computer Science and Engineering with (with Honors) in 2011. Currently Pursuing Doctor of Philosophy (PhD) from Jaypee University of Engineering and Technology, Guna. He has published many research papers in reputed international journals and conferences.



Amit Kumar has completed his M.Tech. in Computer Science and Engineering from Kurukshetra University, Kurukshetra in 2002 and Ph.D. from Jaypee University of Engineering and Technology, Guna in the department of Computer Science and Engineering. The topic of his Ph.D. thesis is Design and Implementation of an Artificial Immune System for Security of Computers. His areas of interest include Artificial Intelligence, Machine Learning, Soft Computing, Data Warehousing, Data-Mining and Computer Immunology. He has published various research papers in international and national journals, presented/published in the national conference, seminars, and workshops. He has attended various national/international conferences, workshops, faculty development programs, seminars etc. He has more than 19 years of teaching experience in various organizations of repute for Ph.D., PG and UG courses of Computer Science Engineering and IT. He is currently supervising Ph.D. in the field of Artificial Intelligence, Machine Learning and Soft Computing.



Neelesh Kumar Jain did his Ph.D. in 2018 from Jaypee University of Engineering and Technology in the department of Computer Science and Engineering. The title of his research was "Efficient Approaches for Digital Image Forgery Detection". He received M.Tech. in Engineering Systems in 2006 from Dayalbagh Educational Institute, Agra and Bachelor's degree in Information Technology in 2003 from Dr. B.R.A. University, Agra. He has been recipient of Director's Gold Medal of Dayalbagh Educational Institute in M.Tech (2006). He has thirteen years of teaching experience for PG and UG courses of Computer Science and Engineering. He has published many research papers in reputed international journals and conferences including SCI indexed journals.