International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

IDO-DKNN: NETWORK ATTACK DETECTION AND MITIGATION VIA OPTIMIZED DEEP KRONECKER NEURAL NETWORK

Waleed Khalid Hussein

College of Biomedical Informatics, University of Information Technology and Communications, Baghdad, Iraq

Email: dr.waleed.khalid@uoitc.edu.iq, waleed8010@gmail.com

Abstract

Complex attack vectors and ever-evolving threats continue to bother network security. Accordingly, effective detection and mitigation approaches have been essential to protect modern digital infrastructure. In turn, this work tries to solve the challenges by means of a comprehensive methodology based on integrating advanced techniques in data preprocessing and feature extraction with an optimized deep learning model. This involves strict preprocessing of the data, which includes cleaning to handle redundancies and missing values, standardization to scale the data uniformly, and the extraction of features based on measures of central tendency, dispersion analyses, and information-based metrics. The significant features selected with Enriched Coati Osprey Algorithm (ECOA) were employed for training Optimized Deep Kronecker Neural Network (Opt-DKNN) -so as to identify the attack's effective detection. This approach improves network security through detecting and mitigating attacks with more appropriate use of advanced feature extraction and optimization techniques. The weighted BIAT model is presented for the mitigation of the attack.

Keywords: *Network security, central tendency measures, Enriched Coati Osprey Algorithm, Optimized Deep Kronecker Neural Network, and Weighted BIAT.*

1. Introduction

The Internet of Things (IoT) has more uses now that smart device expenses have decreased and sensor costs have risen exponentially (Akshaya et al., 2023). In daily life, it is a promising field, much like housing, healthcare, the military, and agriculture. Hackers are becoming more interested in IoT technology as it allows them to leverage their communication skills for a variety of attack types (Palekar and Radhika, 2022). IoT describes how objects are connected to a physical network that is integrated with sensors, software, as well as additional devices to transfer data between devices (Shobana et al., 2022). Because of the improper use of IoT device security flaws, the frequency of cyberattacks and data breaches has skyrocketed across various industries, companies, and enterprises (Samy et al., 2020). Our lives have significantly transformed as a result of the digital revolution, with IoT playing a major role (Vu et al., 2020). Accurately identifying different kinds of Intrusions into IoT networks carried out by zombie hosts under the control of attackers is crucial in the realm of network security (Kanet al., 2021).

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

This often calls for high memory space and network traffic data volume requirements (Popoola et al., 2020). That is, the high feature dimensionality training data transmission and storage would require high network bandwidth and a big memory space, respectively, on an IoT-backed server or Deep Learning Cloud Platform (Popoola et al., 2021). An enormous number of heterogeneous gadgets within the IoT ecosystem make it a very big problem in terms of how to identify them with conventional, rule-based security solutions. Designing the most ideal Safety mechanisms for every kind of gadget is a challenge (Kim et al., 2020). With enhanced data capacity, malware detection has been of critical importance in IoT devices, mainly because the IoT devices are much vulnerable to malware attacks (Riaz et al., 2022). Moreover, simplicity in the protocol makes it vulnerable to online attacks like DDoS attacks and identity thefts (Mahajan et al., 2022).

Continuous wavelet transforms (CWT) image processing in conjunction with deep convolution neural network (CNN) and Internet of Things architecture for resilience energy management and DSM (Elsisi et al., 2023). The attack discovery process makes use of a deep neuro-fuzzy network (DNFN) supported by Elephant Water Cycle (EWC). The Water Cycle Algorithm (WCA)and EHO (Elephant Herd Optimization) are coupled to form the EWC, which is then used to train DNFN (Kumbhar et al., 2024). Using a meta-heuristic optimization approach called the butterfly optimization algorithm, the optimal characteristics for an artificial neural network's learning process are selected (Li et al., 2022). The features selection method FGOA-kNN employs hybrid filter and wrapper selection strategies to identify the most relevant features. According to Taher et al. (2023), the new approach combined with clustering ranks the features and then uses the Grasshopper algorithm (GOA) to reduce the characteristics that rank highest.

CWT-driven While comparing various deep learning and machine learning models, CNN had the highest accuracy. To confirm and show the effectiveness and resilience of the suggested approach under various intensities of hostile attacks, many testing scenarios are provided and carried out. To safeguard the SD-IoT, attack mitigation is done. Recognizing the most important traits and spotting both known and new attacks, improves the proposed model's accuracy and robustness. Memory space needed for data storage was decreased as a result of the LAE method's reduction of the training set's network traffic characteristics' dimensionality. The main contributions of the paper are as follows,

- Utilization of the Enriched Coati Osprey Algorithm (ECOA) for effective feature selection, enhancing the relevance and quality of features used for attack detection.
- Application of the ECOA to fine-tune the Deep Kronecker Neural Network, boosting detection accuracy.
- Deployment of an Weighted BIAT model for effective attack mitigation, ensuring a comprehensive approach to network security.

The structure of the remaining section of the paper is as follows: The previous research papers on the network attack were covered in Section 2; the suggested methodology is explained in detail in Section 3; the results of the suggested methodology are compared with those of alternative techniques in Section 4; and the study is concluded in Section 5.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

2. Literature survey

Devi et al., (2022) proposed a novel, four-phase IoT attack detection mechanism. Pre-processing is the first step in the data normalization process. Then, " higher level statistical characteristics (variance, kurtosis, skewness) are extracted," along with " increased second-order technical indicator-based features (CMF, ATR, CTI, and better EMA) and enhanced Recursive Feature Elimination (RFE)." The CMIHBO approach was also used to select the best characteristics. The outputs of several classifiers, such as " Recurrent Neural Network (RNN), Bidirectional Gated Recurrent Unit (BI-GRU), and Bidirectional Long Short-Term Memory (BI-LSTM)," are then averaged to ascertain the presence of attacks. The Bi-LSTM weights in particular are perfectly modified using Integrated HBO Cat and Mouse (CMIHBO). If an incident is found, " demonstrated mitigation with a BAIT focus" is employed to remove the offending nodes from the networks. Ultimately, numerous strategies are devised to enhance the implemented methodology.

Gotarane et al., (2024) proposed an advanced hybrid architecture that enhances IoT node security by identifying hostile nodes carrying out many types of attacks, including DoS, DDoS, Drop attacks, and Tamper assaults. This is because it integrates DL and whale optimization with a trust index. First, the trust index score for an IoT node is determined by the proposed framework using replay, multiple-max, drop, and temper attacks. Afterwards, it utilizes the Optimized Neural Network model with its trust index score in pinpointing the malignant IoT node with success. In using the Whale Optimization Algorithm, there is the use of a fitness function that establishes ideal weights in the optimization of neural networks.

Wardhani et al., (2023) introduced an innovative model that merges counterfactual and LIME methodologies to give better explanations, along with a blending model for attack categorization. We have run experiments using the newly published datasets of CICIoT2023 and IoTID20, characterizing a very realistic and challenging scenario of problems when intrusion detection is applied on dynamic IoT systems. They correspond to real-time large-scale benchmark datasets concerning attacks against IoT environments.

Qiu et al., (2020) designed a new kind of adversarial attack will be proposed, which can target a DL-based NIDS in IoT environments, such that the DL model in these NIDS is only accessible through black-box access. Two approaches are proposed: 1) with a limited amount of training data, model extraction is used to recreate the model of the black box; 2) a saliency map will then highlight the most relevant features and the contribution of each packet attribute to the detection result, thus enabling us can effectively produce AEs using classical techniques. Using these strategies, we are able to subvert Kitsune- cutting-edge network intrusion detection system.

Alzahrani and Bamhdi (2022) proposed an efficient system designed specially to aid in the identification of attacks using botnets against IoT. This has been realized by the creative fusing of the long short-term memory (LSTM) mechanism with that of a CNN model in recognizing two very common and regarding IoT assaults, BASHLITE, and Mirai, on four different varieties of surveillance cameras. These datasets were collected from live laboratory-connected camera equipment in Internet of Things environments and consist of typical

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

malicious network messages. Based on evaluation measures, the experiment's outcomes showed that the recommended system operated at peak efficiency.

Sagu et al., (2024) suggested a methodology for identifying threats in IoT-enabled cyber-physical systems is developed using two different deep learning models, referred to as hybrid classifiers: Convolutional Neural Network (CNN) and DBN (Deep Belief Network). However, to improve their classification accuracy, DL models must be trained. To adjust the weights of the hybrid classifier, this research also attempts to introduce a new hybrid optimization algorithm known as "Seagull Adapted Elephant Herding Optimization" (SAEHO). Using the feature-extracted dataset as an input, the "Hybrid Classifier + SAEHO" framework categorizes the network as either benign or under assault. Two datasets were compared using sensitivity, precision, accuracy, and specificity. The suggested framework performs better than traditional approaches in every performance indicator.

Ahmim et al., (2023) introduced an identical intrusion detection system based on DL that is intended to be used in an IoT at the Fog or Cloud level. The proposed methodology in terms of DDoS attacks aims to catch each class based on its subclass. This paper combines the different types of DL models, such as LSTM, CNN, Deep Autoencoder, DNN, and others, as its hybrid into one. The proposed architecture has two primary levels. The first one consists of numerous simultaneously trained sub-neural networks for convergence with different kinds of training methods. Each of the frozen outputs of level one along with the starting data serve as input for the second level. Combinations of this type of very different deep neural networks rely on the density of their parameters and play their specific characteristics to achieve extremely high performance.

Vu et al., (2020) proposed a new deep transfer learning approach for learning from data gathered by several unlabeled Internet of Things devices. Concretely, we create A DTL model constructed using two AutoEncoders. First, it trains the first AE, AE 1, in the supervised mode using source datasets (source domains), and then it trains the second AE, AE 2, in the unsupervised mode on target datasets (target domains) without any labelling characteristics. The goal of transfer learning is to match the hidden position of AE 2 (the color of the bottle) with the position of AE 1. The hidden position of AE 2 is then employed to identify attacks in cases that occur in the static domain. We conduct extensive investigations on nine innovative IoT datasets to evaluate the performance of the proposed model.

Sahu et al., (2021) provided an attack detection system and a new security framework that accurately determines rogue devices by filling in the gaps with a Deep Learning model. The recommended method Convolution neural network (CNN) is utilized to get the data's precise illustration of features and then categorizes it using the Long Short-Term Memory (LSTM) Model. Twenty IoT infected with Raspberry Pis comprised the dataset used in the experimental evaluation. The empirical study's accuracy rate in detecting attacks is 96%. Furthermore, it is noted that the suggested model performed better than several recently suggested DL-based attack detection techniques.

Babu et al. (2021) created a unique attack detection structure for the IoT, using the KDD Cup dataset. Primarily, the Euclidean distance and connectedness between the nodes are

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

used to build the potential pathways from node to destination. Also, data transmission uses the shortest route; this route is determined by measuring its minimum distance. The work includes two parts: The first part is done with an Optimized Deep Belief Network previously trained to detect the existence of an attacker. If DBN detects an attacker, then the control is passed to a baiting procedure which removes the node corresponding to that attacker. A newly incoming Whale will perform an algorithm for Distance-based Updates-named W-DU-optimally setting weights in DBN with the aim of assuring an accurate detection procedure.

2.1 Problem statement

Authors Name	**Aim**	**Methods**	**Advantages**	**Limitations**
Devi et al., (2022)	DL-focused strategy to identify and prevent attacks.	RNN, BI-GRU, BILSTM, CMBO	Less memory consumption	• High memory consumption • Difficult to process longer sequences
Gotarane et al., (2024)	To recognize malicious nodes carrying out different types of attacks	WOA, Neural Network	• Less computation usage, • feasible	• Less storage capacity, • expensive
Wardhani et al., (2023)	to improve threat detection and tackle IoT security issues.	MDI	• effective addressing, • Interpretability	High computational and memory resources
Qiu et al., (2020)	To effectively produce AEs with conventional strategies	DNN,	high efficiency,	Less durability
Alzahrani and Bamhdi (2022)	to identify a threat that an IoT platform has intelligently presented.	CNN-LSTM, FFNN, LTM,	• Ability to find new attacks • less time complexity • lower computation cost	Overfitting

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Sagu et al., (2024)	To develop a framework for IoT-enabled cyber-physical systems attack detection	DBN, CNN, SAEHO	Avoids overfitting, supports real-world scenarios	High computational time
Ahmim et al., (2023)	To accurately categorize DDoS attacks of all kinds, including the most similar ones, and innocuous traffic.	CNN, DNN, and LSTM	lowest false alarm rate, highest accuracy	computational complexity
Vu et al., (2020)	To address the "lack of labeled information" problem in popular IoT devices for the model used for training detection	DTL, PCA	Handling large and complex data	Time consumption is high
Sahu et al., (2021)	An attack detection system that effectively detects malicious devices by filling in the gaps with a DL model	CNN, LSTM	• Less time consumption • Avoids overfitting	require large amounts of training data
Babu et al., (2021)	Using the KDD Cup dataset, develop a new attack detection system for the	DBN, W-DU,	Ability to handle large data	Complexity,

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

	Internet of Things.			

3. Proposed methodology

The paper proposes a multivariate approach in the detection and mitigation of network attacks. Feature extraction was done using various measures involving central tendencies, dispersion analyses, and information-based measures such as mutual information and symmetric uncertainty for identifying relevant features and their quantification. This was an extended preprocessing to clean the data by eliminating redundancies, handling missing values, and standardizing variables, hence normalizing their scale. Feature selection will be done by ECOA, which will optimize a subset of features to be used for training. The selected few features are then fed into training the Opt-DKNN, and with further fine-tuning of parameters using ECOA for improved detection accuracy, an Improved Weighted BIAT model for an effective mitigation. It uses the help of this integrated model to achieve robust network security through the detection and mitigation of the attacks with high accuracy. The block diagram of the proposed model is presented in Figure 1.

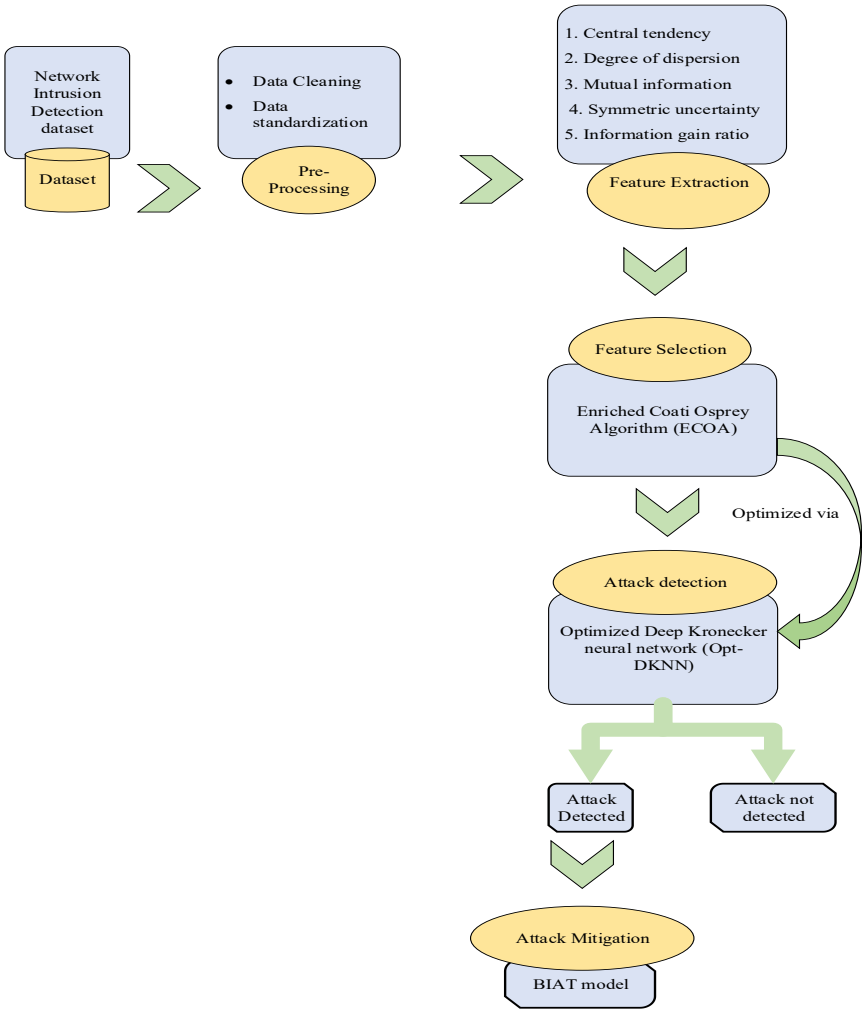

Figure 1: Block diagram of the suggested methodology for detecting attacks

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

3.1. Pre-processing

Pre-processing is one of the major steps in data analysis and machine learning to ready raw data for further processing. Pre-processing would guarantee that this data is standardized, clean, and prepared to enable feature extraction and model training.

3.1.1. Data cleaning

Data cleaning is the process of finding and rectifying errors, inconsistencies, and anomalies in the data collected. This encompasses all procedures that ensure the proper structuring of data, removal of duplicate entries, and checking if all values are present. Other practices that involve cleaning data include interpolation to fill missing gaps or impute values in a set. In filling the gaps, the mean of the previous data and that of the succeeding data are used. Here is the formula:

$$y_i = \frac{y_{i-1}+y_{i+1}}{2} \quad (1)$$

Where the current hour data is denoted by y_i, the previous hour data is represented by y_{i-1}, and the subsequent hour data is represented by y_{i+1}.

3.1.2. Data standardization

Standardization is the most common method used by mathematicians. The usual method transforms various variables into one common scale, with a mean of 0 and a standard deviation of 1. For a generic unit i and variable j, the formula is as follows:

$$y_{ij} = \frac{x_{ij}-M_{x_j}}{S_{x_j}} \quad (2)$$

When variable j's initial value is represented by x_{ij} for unit i, and its mean and standard deviation are denoted by M_{x_j} and S_{x_j}, respectively,

3.2. Feature Extraction

Feature extraction is an important phase in data processing; it transforms raw data into a set of useful features applicable for model training. The goal here is to decrease the data dimensionality while maintaining most of its significant attributes to improve the efficiency and accuracy of machine learning models.

3.2.1. Central tendency

The measures of central tendency, such as mean and median, describe the typical value or central position of a feature in the dataset.

$$Mean = \frac{Sum\ of\ all\ values}{Total\ number\ of\ values} \quad (3)$$

$$Median = \begin{cases} \frac{l+1}{2}, & for\ odd \\ \frac{l}{2}, & for\ even \end{cases} \quad (4)$$

Where l is the last value.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

3.2.2. Degree of dispersion

It can be used to ascertain the degree of dispersion, about how much each data point in a dataset differs. Diversity or variability of the data points taken from the central tendency can be measured. It considers variance for each feature, standard deviation, and IQR.

$$Variance = \frac{\sum(x_i-\mu)^2}{N} \quad (5)$$

$$SD\ (\sigma) = \sqrt{\frac{\sum(x_i-\mu)^2}{N}} \quad (6)$$

Where x_i is the current value and μ is the mean value.

❖ **IQR**

Variability is the property of the data that defines and divides the dataset into quartiles, or regions within which most of the data is contained. Quartiles divide the data into equal halves of the dataset, or first, second, and third, and finally, fourth quartiles. These three quartile borders, which divide these halves, are symbolized by SQ1, SQ2, and SQ3, respectively. SQ1 is the median value of the first half, while SQ2 is the middle value of the second half. The interquartile range has the following Eq. (7):

$$IQR = SQ3 - SQ1 \quad (7)$$

3.2.3. Mutual information

The following is the equation for Mutual Information (MI) between two discrete random variables, $X\ and\ Y$:

$$I(X;Y) = \sum_{x\in X}\sum_{y\in Y} p(x,y)\log\left(\frac{p(x,y)}{p(x)p(y)}\right) \quad (8)$$

The joint probability distribution of $X\ and\ Y$ is represented by $p(x,y)$, which is the probability that $X = x$ and $Y = y$ simultaneously. The marginal probability distribution of X, written $p(x)$, is the probability that $X = x$ without regard to the value of Y. Likewise, the marginal probability distribution of Y is $p(y)$, is the probability that $Y = y$ without regard to the value of X.

3.2.4. Symmetric uncertainty

Symmetric uncertainty is the normalized measure of dependency between two input variables, and it normalizes mutual information by considering entropies of individual variables. That normalizes the measure so that it is symmetric, that is, it does not bias one variable over another and views the relationship between the variables as mutual.

$$SU(X,Y) = 2 \times \frac{I(X;Y)}{H(X)+H(Y)} \quad (9)$$

The symmetric uncertainty and mutual information between $X\ and\ Y$ are defined by the variables: $SU(X,Y)$ and $I(X;Y)$, respectively. $H(X)$ denotes entropy of X, a quantification of

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

uncertainty or amount of information needed to describe X. $H(Y)$ denotes entropy of Y, a quantification of uncertainty or amount of information needed to describe Y.

3.2.5. Information gain ratio (IGR)

The IGR is a statistic used to decide which attribute is best to split the data. Being an update over the normalized version of the Information Gain, it considers the entropy or intrinsic information of the attribute. Attributes with a high number of unique values, which would otherwise dominate the decision tree, are penalized by the information gain ratio.

$$IGR(X,Y) = \frac{IG(X,Y)}{H(X)} \qquad (10)$$

Where $IGR(X,Y)$ is the Information Gain Ratio between attribute X and target variable Y, $IG(X,Y)$ is the Information Gain between $X\ and\ Y$, calculated as: $IG(X,Y) = H(Y) - H(Y \mid X), H(Y)$ is the entropy of the target variable Y, $H(Y \mid X)$ is the conditional entropy of $Y\ given\ X$, representing the remaining uncertainty about Y after knowing X and $H(X)$ is the Entropy of the attribute X, representing the intrinsic information of X.

3.3. Feature Selection using Enriched Coati Osprey Algorithm (ECOA)

The feature selection stage receives the extracted features from various data. The best features are selected using the ECOA algorithm from the retrieved features. The combined COA and OOA optimization methods are known as the ECOA. In order to enhance performance, the coati's behavior and the defense strategy from the PFA optimization are hybrid.

Using the ECOA algorithm technique, coatis are seen as members of the population in this population-based metaheuristic method. The chosen factors are determined by each coati's location within the search zone. Coati's inclusion by the ECOA thus offers a viable solution to the problem. Before the ECOA implementation begins, the coatis' location in the search space is randomly updated using Equation (11).

$$X_i: x_{i,j} = lb_j + r.\left(ub_j - lb_j\right), i = 1,2,\ldots..,N\ and\ j = 1,2.,..,m \qquad (11)$$

In the given scenario, X_i represents the position of the i^{th} coati within the search space, m indicates the total number of decision variables, $x_{i,j}$ indicates the value of the j^{th} decision variable, N indicates the number of coatis, r is a random real value within the range [0, 1], and lb_j and ub_j indicate the lower and upper bounds of the j^{th} decision variable, respectively. The ECOA coati community is represented by the population matrix, which can be expressed mathematically as the following matrix X.

$$X = \begin{bmatrix} X_1 \\ \vdots \\ X_i \\ \vdots \\ X_N \end{bmatrix}_{N\times m} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,d} & \cdots & x_{1,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{i,1} & \cdots & x_{i,d} & \cdots & x_{i,m} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ x_{N,1} & \cdots & x_{N,d} & \cdots & x_{N,M} \end{bmatrix}_{N\times m} \qquad (12)$$

The intended function of the problem is evaluated at various values when potential solutions are grouped in decision factors. Utilizing Eq. (13), these values are displayed.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

$$F = \begin{bmatrix} F_1 \\ \vdots \\ F_i \\ \vdots \\ F_N \end{bmatrix}_{N\times 1} = \begin{bmatrix} F(X_1) \\ \vdots \\ F(X_i) \\ \vdots \\ F(X_N) \end{bmatrix}_{N\times 1} \quad (13)$$

The obtained objective variable F_i is measured by the i^{th} coati, and the vector F_i represents the attained goal function. If metaheuristic methods such as the one proposed for ECOA are used, the value of the target function provides a standard for potential solution quality. So, the best member of the population is the one whose evaluation yields the highest value for the target function. Upon iterating through the candidate solutions, the algorithm updates the most qualified individual in the population.

i.Mathematical model of COA

Two of the coatis' natural behaviors are modeled before the coatis (candidate solutions) are moved within the ECOA. These behaviors include (i) the ways in which coatis attack and (ii) the ways in which they flee from potential predators. The ECOA population is updated twice a year as a result.

Phase 1: Exploration phase

Using a computational model of their prey-attacking techniques, we first update the coati density in the search zone. A few coatis climb the tree to get close to and threaten a victim. As the prey lands, other coatis congregate around it, anticipating its arrival behind a tree. The coati's assault and pursue the victim as soon as they touch down. As indicated by Eq. (14) coatis can go to various locations inside the search zone with this strategy, demonstrating the ECOA's ability to conduct an exhaustive search in the domain of problem-solving.

$$X_i^{P_1}: x_{i,j}^{P_1} = x_{i,j} + r.\left(Prey_j - I.x_{i,j}\right) \quad (14)$$

$for\ i = 1,2,\ldots,\left(\frac{N}{2}\right)$ and j=1,2,….,m. The prey is spread out at random throughout the search area after being released to the ground. Using Eq. (15) and Eq. (16) to replicate the search region and this arbitrary point as a guide, ground-based coatis move.

$$Prey^G: Prey_j^G = lb_j + r.(ub_j - lb_j) \quad (15)$$

$$X_i^{P_1}: x_{i,j}^{P_1} = \begin{cases} x_{i,j} + r.\left(Prey_j^G - I.x_{i,j}\right), & F_{Prey^G} < F_i \\ x_{i,j} + r.\left(x_{i,j} - Prey_j^G\right), & else \end{cases} \quad (16)$$

For $i = \left[\frac{N}{2}\right] + 1, \left[\frac{N}{2}\right] + 2, \ldots., N\ and\ j = 1,2, \ldots., m$

The search agent stays in the original place until each coati's updated position increases the value of the objective function; at that point, the update procedure is going to accept it. The topic of this update condition is the simulated values of $i = 1,2,\ldots,N$ provided by Eq. (17).

$$X_i = \begin{cases} X_i^{P_1}, & if\ F_i^{P_1} < F_i \\ X_i, & Otherwise \end{cases} \quad (17)$$

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Where $X_i^{P_1}$ represents the updated position of the coati at the i^{th} dimension, $F_i^{P_1}$ represents its fitness value, and r is a random variable that falls between [0, 1]; this is all based on the ECOA exploitation phase.

The prey's location in the search area is actually represented by the iguana, which also symbolizes the best member's position. Its j^{th} dimension is $Prey_j$, and an integer i is randomly selected from the interval [1, 2]. A variable called $Prey^G$ and indicates the prey's randomly generated place on Earth. The dimension of $Prey_j^G$ is its j^{th} dimension; the floor function is $\lfloor \cdot \rfloor$; the objective function's value is F_{Prey^G}.

Phase 2: Exploitation phase

The second stage of Coatis' position is updated using the exploitation phase of the Osprey Optimization Algorithm (OOA). Here, coatis emote in imitation of fleeing from predator assaults, signifying ECOA's ability to use local search tactics to establish a base close to their present position. Using Eq. (18) to Eq. (21) a random position is created close to each coati's current location in order to replicate this behavior.

$$lb_j^{local} = \frac{lb_j}{t}, ub_j^{local} = \frac{ub_j}{t}, where\ t = 1,2, \dots, T \tag{18}$$

$$X_i^{P_2}: x_{i,j}^{P_2} = x_{i,j} + \frac{\left(lb_j^{local} + r.\left(ub_j^{local} - lb_j^{local}\right)\right)}{t} \tag{19}$$

$$x_{i,j}^{P_2} = \begin{cases} x_{i,j}^{P_2}, lb_j^{local} \le x_{i,j}^{P_2} \le ub_j^{local} \\ lb_j^{local}, x_{i,j}^{P_2} < lb_j^{local} \\ ub_j^{local}, x_{i,j}^{P_2} > ub_j^{local} \end{cases} \tag{20}$$

Where $i = 1,2, \dots, N$ and j=1,2,….,m and $t = 1,2, \dots, T$.

$$X_i = \begin{cases} X_i^{P_2}, if\ F_i^{P_2} < F_i \\ X_i, else \end{cases} \tag{21}$$

Where $X_i^{P_2}$ is the updated position of the coati at i^{th} dimension, $F_i^{P_2}$ is its fitness value, using the exploitation phase of ECOA, $x_{i,j}^{P_2}$ is its j^{th} dimension, lb_j^{local} and ub_j^{local} are the lower and upper bound of the j^{th} local decision variable respectively, lb_j and ub_j are the lower bound, t is the iteration count, and upper bound of the j^{th} decision variable, respectively.

3.4. Attack detection using Opt-DKNN

The weight matrices are constructed via Deep Kronecker neural networks (DKNN), which make use of the Kronecker product. These findings show that neural networks with adaptive activation functions can be broadly modeled using KNNs, and numerous pre-existing neural networks can be transformed into specific cases of KNNs. In actuality, the general adaptive activation function of the KNN is the same as that of the traditional Feed-Forward Neural Networks (FNNs), taking the following form:

$$\phi_{\alpha,\omega}(x) = \sum_{k=1}^{K} \alpha_k\, \phi_k(\omega_k x), K \in \mathbb{N}_{\ge 1}, \alpha = (\alpha_k), \omega = (\omega_k) \tag{22}$$

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Where α_k denotes potentially trainable or fixed parameters and ϕ_k's represent fixed activation functions. Therefore, 18 it is not necessary to compute the Kronecker product in order to implement the KNN. Nonetheless, with nearly the same 20 parameters, a far broader network can be built using the 19 Kronecker product than with an FNN.

A FNN with depth D_p is a function that is defined by combining several layers, including an input layer, $D_p - 1$ represents the hidden layers, and an output layer. There are N_l neurons in the l^{th} hidden layer. An affine transformation is applied in the previous layer, where each hidden layer obtains an output $z^{l-1} \in \mathbb{R}^{N_{l-1}}$.

$$\mathcal{L}_l(z^{l-1}) \triangleq W^l z^{l-1} + b^l \tag{23}$$

The weight matrix in this case is $W^l \in \mathbb{R}^{N_l \times N_{l-1}}$, and the bias vector connected to the l^{th} layer is $b^l \in \mathbb{R}^{N_l}$. Before being sent as an input to the next layer, each component of the converted vector is subjected to a nonlinear activation function $\varphi_1(\cdot)$. Following an output layer comes the activation function, which is an identity function. Consequently, the neural network's final representation is provided by,

$$u^{FF}(z) = (\mathcal{L}_D \circ \phi_1 \circ \mathcal{L}_{D-1} \circ \ldots \circ \phi_1 \circ \mathcal{L}_l)(z) \tag{24}$$

The composition operator is represented by the operator $\circ$. The trainable parameters in the network are represented by $\Theta_{FF} = \{W^l, b^l\}_{l=1}^{D}$. Given a vector $v = [v_1, \cdots, v_n]^T \in \mathbb{R}^{N_l}$, let's examine the different norms of v:

$$\|v\|_1 = \sum_{i=1}^{n} |v_i|, \quad \|v\|^2 = \sum_{i=1}^{n} v_i^2, \ \|v\|_\infty = \max_{1 \le i \le n} |v_i| \tag{25}$$

The n^{th} greatest singular value of a matrix $M \in \mathbb{R}^{m \times n}$, where $m \geq n$, is denoted by $\sigma_{\min(M)}$. Furthermore, the descriptions of the Frobenius standard and the spectral standard are,

$$\|M\| = \max_{\|x\|=1} \|Mx\|, \ \|M\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} M_{ij}^2 \tag{26}$$

where M's (i, j)- component is represented by M_{ij}. Let $1_{s \times t}$ be the size $s \times t$ matrix with all 1s elements.

3.4.1. DKNN

Let K represent a positive integer. The l^{th} block weight matrix and block bias vector of a FNN with parameters $\Theta_{FF} = \{W^l, b^l\}_{l=1}^{D}$ can be defined as follows:

$$1_{K \times K} \otimes W^l = \begin{bmatrix} W^l & \ldots & W^l \\ \vdots & \ddots & \vdots \\ W^l & \ldots & W^l \end{bmatrix} \in \mathbb{R}^{N_l K \times N_{l-1} K}, \ 1_{K \times 1} \otimes b^l = \begin{bmatrix} b^l \\ \vdots \\ b^l \end{bmatrix} \in \mathbb{R}^{N_l K} \tag{27}$$

The Kronecker product is denoted as $\otimes$. Consider that the block-wise block activation function $\vec{\phi}$. In other words, let $z = [z_1, \cdots, z_K]^T \in \mathbb{R}^{nK}$ for $z_j \in \mathbb{R}^{nK}$ for $1 < j \leq K$.

$$\vec{\phi}(z) = \begin{bmatrix} \phi_1(z_1) \\ \vdots \\ \phi_K(z_K) \end{bmatrix} \tag{28}$$

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

When activation functions are applied element-by-element (φ_j's). Following that, it constructs a neural network in the way detailed below by utilizing the block weight matrices and block bias vectors. For example, $z^0 = z$ would be the input, and $z^l = (1_{K\times K} \otimes W^l)(1_{K\times 1} \otimes z^0) + 1_{K\times 1} \otimes b^l. If\ 2 \le l < D_p$,

$$z^l = (1_{K\times K} \otimes W^l)\vec{\phi}(z^{l-1}) + 1_{K\times 1} \otimes b^l \qquad (29)$$

Furthermore, $z^{D_p} = (1_{K\times K} \otimes W^{D_p})\vec{\phi}(z^{D_p-1}) + b^{D_p}$. With KNN neurons at the l^{th} layer, z^{D_p} is a D_p-layer FNN with the same number of network parameters as u^{FF} in Eq. (3). $\omega^l, \alpha^l \in \mathbb{R}^K$ are the scaling parameters introduced to appropriately scale the block weight matrices and the block bias vectors. The row vector in this case is α^l, and the column vector is ω^l. Then find the block bias vectors and scaled block weight matrices using Eq. (30).

$$\widetilde{W}^l = (\omega^l \otimes \alpha^l) \otimes W^l, \tilde{b}^l = \omega^l \otimes b^l, 1 \le l < D_p$$

$$\text{And } \widetilde{W}^D = 1_{1\times K} \otimes W^{D_p}, \tilde{b}^{D_p} = b^{D_p} \qquad (30)$$

For $1 \le l \le D$, let

$$z^l \coloneqq \tilde{\mathcal{L}}_l(z^{l-1}) = \widetilde{W}^l z^{l-1} + \tilde{b}^l \qquad (31)$$

Next, acquire the representation provided by

$$u_\Theta^K(z) = (\tilde{\mathcal{L}}_{D_p} \circ \vec{\phi} \circ \tilde{\mathcal{L}}_{D_p-1} \circ \ldots \circ \vec{\phi} \circ \tilde{\mathcal{L}}_l)(1_{K\times 1} \otimes z) \qquad (32)$$

This form is known as a KNN. The network parameters are as follows: $\Theta_{\mathrm{K}} = \{W^l, b^l\}_{l=1}^{D_p} \cup \{\omega^l, \alpha^l\}_{l=1}^{D_p}$.

It is observed that each hidden layer of the KNN has K times more neurons than the FNN. Nevertheless, the Kronecker product only causes a $2K(D-1)$ difference in the total number of parameters. Additionally, as seen below, the KNN can be understood as a novel class of neural networks that generalizes an existing class of FNN, specifically to make use of adaptive activation functions.

It is possible to create the Kronecker network effectively without building the block bias vectors $\{\tilde{b}^l\}_l$ and block weight matrices $\{\widetilde{W}^l\}_l$. It is possible to verify that the composition can represent the Kronecker neural network.

$$u^K(z) = (\mathcal{L}_D \circ \tilde{\phi}^{D-1} \circ \mathcal{L}_{D-1} \circ \ldots \circ \tilde{\phi}^1 \circ \mathcal{L}_l)(z) \qquad (33)$$

Where the trainable parameters $\{\omega^l, \alpha^l\}$ determine the activation function at the l^{th} layer, which is no longer deterministic.

$$\tilde{\phi}^l(\mathcal{L}_1(z); \omega^l, \alpha^l) = \sum_{k=1}^K \alpha_k^l \phi_K(\omega_k^l \mathcal{L}_1(z)), l = 1, .., D-1 \qquad (34)$$

In Figure 2, a three-layer Kronecker neural network schematic is displayed. The number of units in the Kronecker Dense layers of the DKNN are optimized using the ECOA algorithm.

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

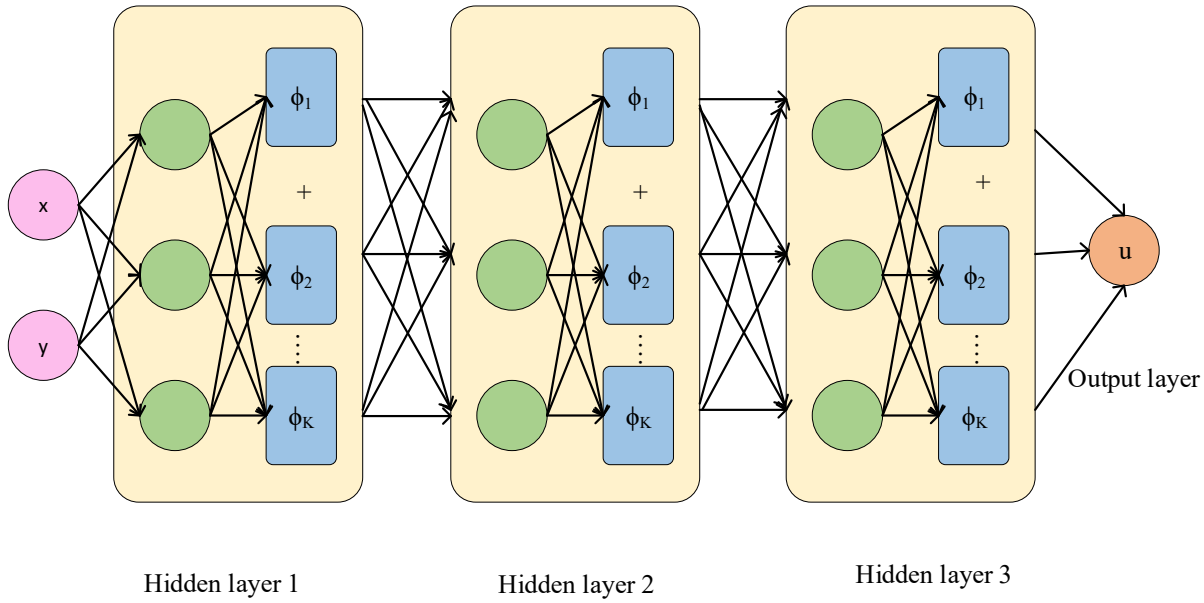

Figure 2: Layered architecture of the DKNN model

3.5. Mitigation using Weighted BIAT

The position of the intruder within the network is ascertained using the BAIT technique. The three primary stages of the BAIT network are route maintenance, intrusion detection and mitigation, and discovery.

i) Data Request from Source to Destination

A data request is sent to a destination node or device in a network communication scenario by a source node or device. Usually, a variety of network protocols are used to send this request across the network. Establishing a communication channel between the source and destination for the exchange of data is the fundamental notion behind the data request process, however the specifics may differ based on the protocol being utilized.

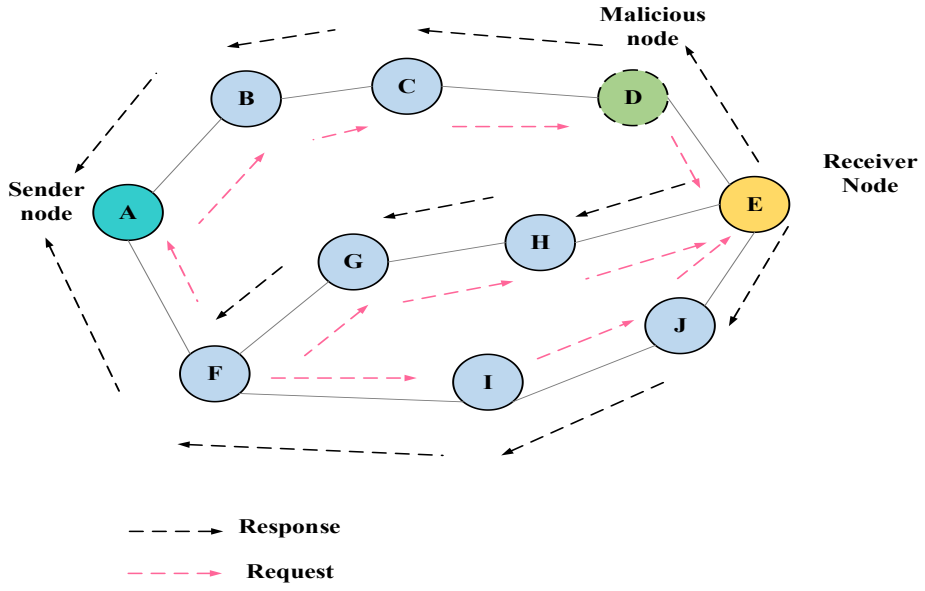

Figure 3: BAIT based malicious node mitigation using shortest route selection

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

ii) **Acknowledgement from Shortest Path**

The network chooses the most effective route for data routing from the source to the destination when using a routing algorithm, like the shortest path algorithm. The source node sends the data packets along the shortest path after it has been found. In order to verify that the packets were successfully received by the source, intermediate nodes along the path can send acknowledgments back to it as they move across the network.

iii) Data Sensitivity

The term "data sensitivity" describes the degree of confidentiality or significance attached to the sent data. There may be differences in the sensitivity of different kinds of data. In contrast to non-sensitive data, such public information or non-personal data, sensitive financial information or personal data, for instance, may need more robust security measures.

iv) Weight Function

A weight function used by a routing algorithm assigns a numerical weight, to the links or edges that connect the nodes. Such weights mostly represent parameters like cost, delay, bandwidth, or distance. The weight function assists in determining the best course of action for meeting predetermined selection criteria. In other words, for any path that is of lesser length or faster pace, the weight assigned is smaller. Computing Trust by Several Shortest Paths.

Where there are multiple shortest paths from source to destination, each may be evaluated against a calculated trust. One way to calculate the trust would be based on variables such as the conditions of the network, past history, security measures, or the reputation of the nodes or links in each path. These variables can be combined using mathematical models, including Bayesian networks, to calculate a trust score-a weighted Bait model-for every path. This is a more complex mathematical formula that incorporates attacker mitigation with the weighted Bait model.

$$T(P_i) = \alpha * F_1(P_i) + \beta * F_2(P_i) + \gamma * F_3(P_i) + \cdots . + \eta * F_n(P_i) - \omega * A(P_i) \quad (34)$$

Where, $T(P_i)$ is the trust score for the i^{th}-shortest path P_i. F_1, F_2, F_3, ..., F_n are the contributing factors to the calculation of trust and $A(P_i)$ is the attacker mitigation factor for path P_i. $\alpha, \beta, \gamma, ., \eta$ are weights assigned to the factors and ω is the weight assigned to the attacker mitigation factor.

The attacker mitigation factor $A(P_i)$, characterizes how well the path P_i mitigates attacks. It may consider a variety of security measures taken on the path, such as intrusion detection systems, firewalls and encryption mechanisms. The attacker mitigation factor $A(P_i)$ can have a binary value (0 or 1) indicating whether or not effective mitigation measures are in place:

4. Result and discussion

In this part, the outcomes of the suggested model are contrasted with those of the current methods. The author developed the dataset with fertilizer recommendations. The Python platform is used for the implementation. Of the total data, thirty percent are used for testing and seventy percent are used for training. To train on a single subset, the model needs to be

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

partitioned. After that, its predictability is assessed by analyzing how well it performs on the other subset.

4.1. Dataset Description

The Network Intrusion Detection dataset is used for implementation. This dataset consists of TCP/IP connection records that were gathered from a military network environment simulation that was especially created to mimic an attack on a US Air Force local area network. It contains raw dump data from many simulated incursions; each record corresponds to a connection, which is specified as a series of TCP packets exchanged over a predetermined amount of time between a source and a target IP address. Every connection is classified as "Normal" or "Anomalous," the latter of which includes many specialized attack kinds. Each relationship in the dataset has 41 attributes: 38 quantitative (numerical) and 3 qualitative (descriptive). About 100 bytes make up each record, which contains comprehensive data to help with network intrusion detection and analysis(SAMPADA BHOSALE, (2018)**)**.

4.2. Comparison of the proposed attack detection model

This section compares the suggested AMNN model to other methods that are existing, such as Bi-LSTM, CNN, DBN, and AE. Table 1 presents the comparison.

Table 1: Performance metrics for 30% of testing and 70% of training

Classifier	**Bi-LSTM**	**CNN**	**DBN**	**AE**	**Proposed**
Accuracy	0.8916	0.9127	0.9106	0.9035	0.9886
Specificity	0.8985	0.9084	0.9172	0.9005	0.9875
Sensitivity	0.8857	0.9164	0.9048	0.9062	0.9896
Precision	0.9093	0.9200	0.9263	0.9128	0.9891
F1 Score	0.8974	0.9182	0.9154	0.9095	0.9894
FNR	0.1143	0.0836	0.0952	0.0938	0.0104
FPR	0.1015	0.0916	0.0828	0.0995	0.0125
NPV	0.8724	0.9043	0.8934	0.8931	0.9880
MCC	0.7829	0.8246	0.8208	0.8063	0.9771

The table compares the performance metrics of different classifiers, including Bi-LSTM, CNN, DBN, AE, and the Proposed method, based on a dataset split with 70% for training and 30% for testing. Compared to the other classifiers, the proposed technique performs better on all measures. With an accuracy of 0.9886, it reaches the maximum. It receives a sensitivity of 0.9896 and a specificity of 0.9875. Both accuracy (0.9891) and F1 Score (0.9894) are excellent in the proposed strategy. Furthermore, it records the lowest False Positive Rate (FPR) of 0.0125 and False Negative Rate (FNR) of 0.0104. With values of 0.9880 and 0.9771, respectively, the

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Proposed approach also has the highest Negative Predictive Value (NPV) and Matthew's Correlation Coefficient (MCC).

Table 2: Performance metrics for 20% of testing and 80% of training

Classifier	Bi-LSTM	CNN	DBN	AE	Proposed
Accuracy	0.9105	0.8891	0.8944	0.9014	0.9944
Specificity	0.9108	0.8922	0.8913	0.9057	0.9945
Sensitivity	0.9102	0.8863	0.8972	0.8975	0.9944
Precision	0.9202	0.9029	0.9032	0.9150	0.9951
F1 Score	0.9152	0.8945	0.9002	0.9062	0.9948
FNR	0.0898	0.1137	0.1028	0.1025	0.0056
FPR	0.0892	0.1078	0.1087	0.0943	0.0055
NPV	0.8997	0.8741	0.8846	0.8866	0.9937
MCC	0.8205	0.7777	0.7882	0.8024	0.9888

The proposed approach, CNN, DBN, AE, and Bi-LSTM are evaluated in the table with 80% training and 20% testing data. The Proposed technique achieves the best accuracy of 0.9944, outperforming the other methods. It has 0.9945 and 0.9944 for specificity and sensitivity, respectively. Additionally, it performs best in terms of precision (0.9951) and F1 Score (0.9948), indicating a well-rounded effort. With a FNR of 0.0056 and an FPR of 0.0055, the Proposed technique exhibits the lowest errors. The proposed technique outperforms the previous classifiers in terms of overall predictive power, with an NPV of 0.9937 and an MCC of 0.9888.

Accuracy

The Proposed technique consistently produces the highest accuracy in both cases, as can be shown by comparing the accuracy values from Tables 1 and 2. The accuracy of the suggested approach is 0.9886 in Table 1, while it increases to 0.9944 in Table 2.

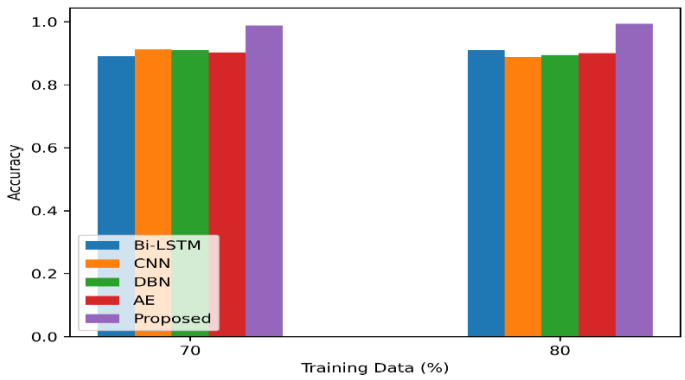

Figure 4: Graphical comparison of Accuracy

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

Precision

The Proposed technique consistently performs better when comparing the precision values between Tables 1 and 2. The Proposed approach obtains an accuracy of 0.9891 in Table 1. The precision increases to 0.9951 in Table 2. This improvement in precision with a bigger training set suggests that more data enhances the Proposed method's capacity to accurately identify affirmative cases.

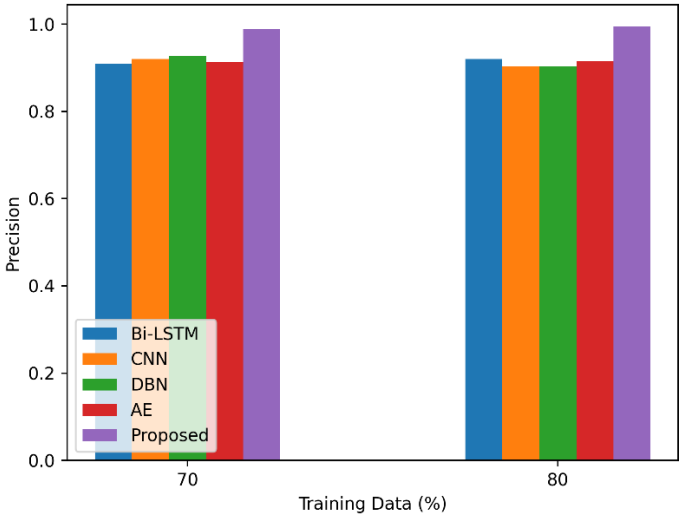

Figure 5: Graphical comparison of Precision

F1- score

The Proposed approach consistently outperforms the other classifiers, as demonstrated by a comparison of the F1 Score values between Tables 1 and 2. The proposed method's F1 Score in Table 1 is 0.9894. The F1 Score in Table 2 drops to 0.9948 somewhat. This variation implies that the Proposed approach produces even better F1 Scores with additional training data, even while it maintains strong performance across different splits.

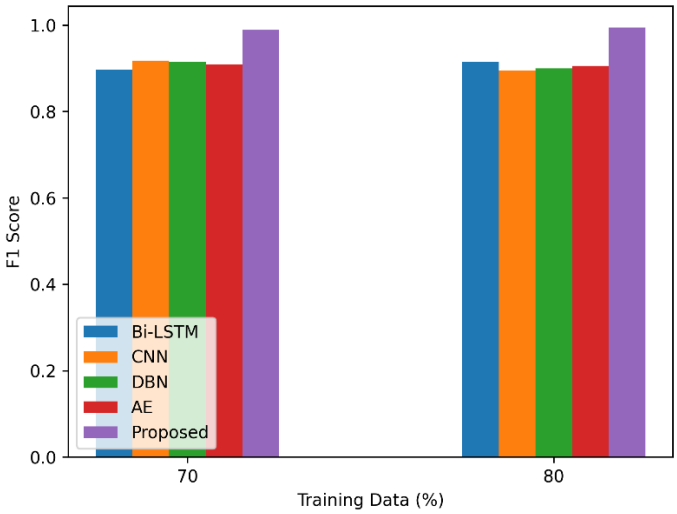

Figure 6: Graphical comparison of F-Measure

Specificity

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

When the specificity values from Tables 1 and 2 are compared, it is clear that the Proposed approach performs better with more training data. The proposed method's specificity in Table 1 is 0.9875. In Table 2, this value rises to 0.9945. The rise in specificity shows that when more training data is used, the proposed technique gets better at correctly recognizing negative cases.

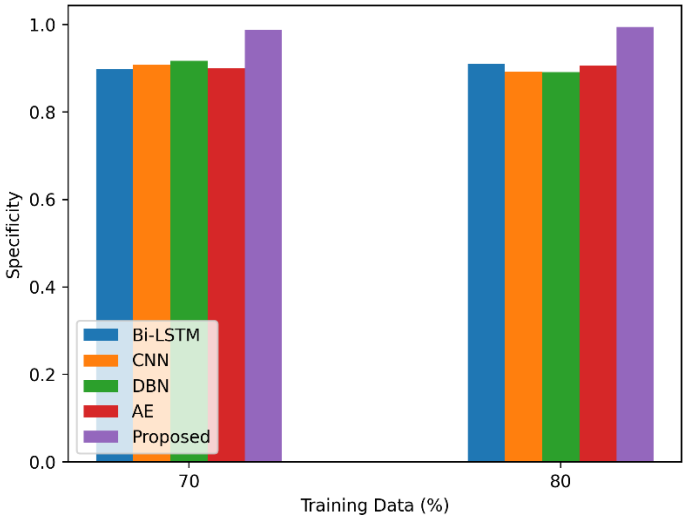

Figure 7: Graphical comparison of specificity

Sensitivity

More training data significantly improves the proposed technique, as seen by the comparison of the sensitivity values from Tables 1 and 2. The proposed method's sensitivity is 0.9896 in Table 1. In Table 2, this value marginally drops to 0.9944. The rise in sensitivity suggests that as training data volume increases, the suggested technique gets better at correctly recognizing positive cases.

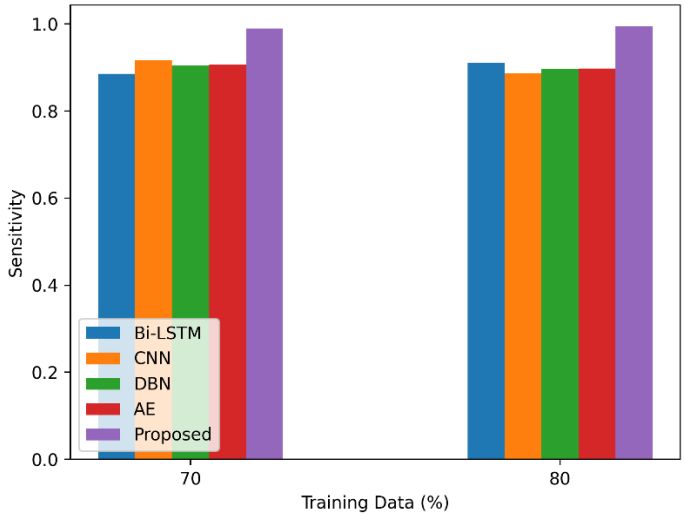

Figure 8: Graphical comparison of sensitivity

MCC

When comparing the MCC values across Tables 1 and 2, the Proposed approach clearly benefits from having more training data. The proposed method's MCC in Table 1 is 0.9771. In Table 2,

Received: August 05, 2025

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

this value increases to 0.9888. As additional training data is used, the increase in MCC suggests that the proposed technique performs better overall in balancing true positive and true negative rates.

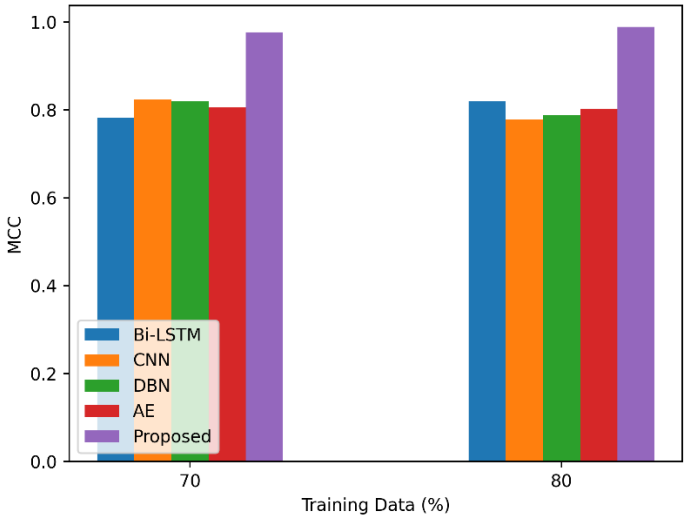

Figure 9: Graphical comparison of MCC

NPV

Table 1 and Table 2's NPV values can be compared to see how much the Proposed technique has improved with additional training data. The NPV of the proposed technique is 0.9880 in Table 1. In Table 2, this value rises to 0.9937. The increase in NPV shows that more training data increases the Proposed method's accuracy in identifying negative instances.

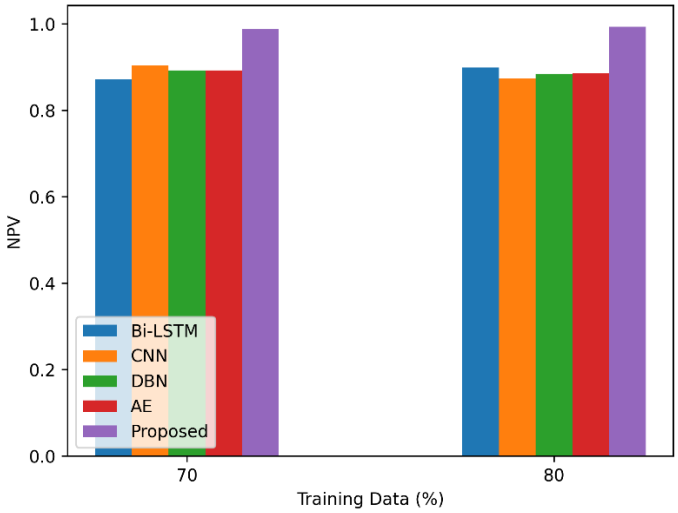

Figure 10: Graphical comparison of NPV

FPR

The proposed method demonstrates a considerable reduction with greater training data when comparing the FPR values from Tables 1 and 2. The proposed method's FPR is 0.0125 in Table 1. Table 2 shows that this value drops to 0.0055. As more training data is gathered, the

International Journal of Applied Mathematics
Volume 38 No. 4s, 2025
ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

suggested strategy becomes more successful at reducing false positives, as evidenced by the decline in FPR.

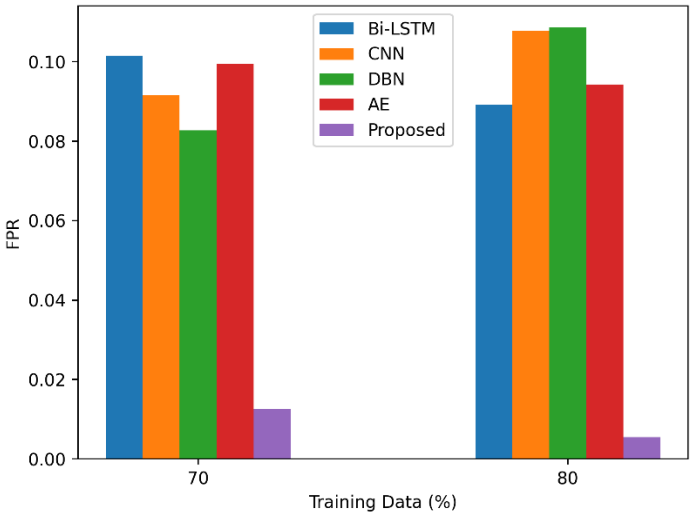

Figure 11: Graphical comparison of FPR

FNR

When comparing the FNR values in Tables 1 and 2, it is evident that the Proposed approach performs better with more training data. The proposed method's FNR in Table 1 is 0.0104. In Table 2, this value drops to 0.0056. The decrease in FNR suggests that as training data volume rises, the suggested approach gets better at reducing false negatives.

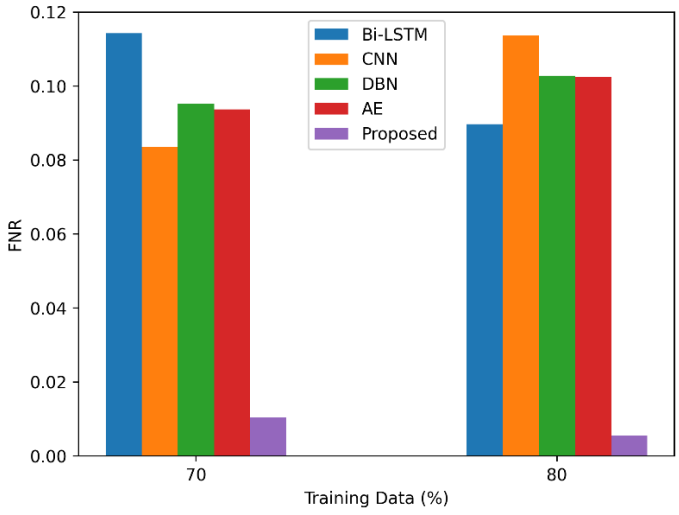

Figure 12: Graphical comparison of FNR

5. Conclusion

In conclusion, the proposed ECOA-DKNN methodology was utilized for enhanced network security in the face of cyber threats that are increasingly complex. Ensuring quality input data feeds into the subsequent analysis, comprehensive pretreatment data approaches-like cleaning, standardization, and advanced feature extraction by dispersion analysis, central tendency measurements, and information-based metrics-form part of the methodology. Only the most

Received: August 05, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

relevant data is fed in to train the Opt-DKNN, and that is definitely one desirable consequence of the crucial functioning of the ECOA in exact identification of the important features. This optimization yields a highly effective and precise detection mechanism that can detect attack patterns that are even the most complicated. It achieves this by the fine-turning of DKNN with ECOA. Also, inbuilt Weighted BIAT model for mitigation of an attack provides second-layer defense that ensures the threat is identified as well as eliminated effectively. Its comprehensive approach, using sophisticated feature selection and optimization of neural networks along with mitigation techniques for various attacks, provides strong defense mechanisms against dynamic cyber threats.

Reference

1. Ahmim, A., Maazouzi, F., Ahmim, M., Namane, S. and Dhaou, I.B., 2023. Distributed denial of service attack detection for the Internet of Things using hybrid deep learning model. IEEE Access, 11, pp.119862-119875.
2. Akshaya, V., Mandala, V., Anilkumar, C., VishnuRaja, P. and Aarthi, R., 2023. Security enhancement and attack detection using optimized hybrid deep learning and improved encryption algorithm over the Internet of Things. Measurement: Sensors, 30, p.100917.
3. Alzahrani, M.Y. and Bamhdi, A.M., 2022. Hybrid deep-learning model to detect botnet attacks over Internet of Things environments. Soft Computing, 26(16), pp.7721-7735.
4. Babu, M.R. and Veena, K.N., 2021. Implementing optimized classifier for distributed attack detection and BAIT-based attack correction in IoT. International Journal of System Assurance Engineering and Management, pp.1-16.
5. Brindha Devi, V., Ranjan, N.M. and Sharma, H., 2022. IoT attack detection and mitigation with optimized deep learning techniques. Cybernetics and Systems, pp.1-27.
6. Elsisi, M., Su, C.L. and Ali, M.N., 2023. Design of reliable IoT systems with deep learning to support resilient demand side management in smart grids against adversarial attacks. IEEE Transactions on Industry Applications.
7. Gotarane, V., Abimannan, S., Hussain, S. and Irshad, R.R., 2024. A Hybrid Framework Leveraging Whale Optimization and Deep Learning with Trust-Index for Attack Identification in IoT Networks. IEEE Access.
8. Kan, X., Fan, Y., Fang, Z., Cao, L., Xiong, N.N., Yang, D. and Li, X., 2021. A novel IoT network intrusion detection approach based on adaptive particle swarm optimization convolutional neural network. Information Sciences, 568, pp.147-162.
9. Kim, J., Shim, M., Hong, S., Shin, Y. and Choi, E., 2020. Intelligent detection of IoT botnets using machine learning and deep learning. Applied Sciences, 10(19), p.7009.
10. Kumbhar, K. and Mukherji, P., 2024. An optimized deep strategy for recognition and alleviation of DDoS attacks in SD-IoT. Network: Computation in Neural Systems, pp.1-32.
11. Li, Y., Ghoreishi, S.M. and Issakhov, A., 2022. Improving the accuracy of network intrusion detection systems in medical IoT systems through butterfly optimization algorithm. Wireless Personal Communications, 126(3), pp.1999-2017.
12. Mahajan, N., Chauhan, A., Kumar, H., Kaushal, S. and Sangaiah, A.K., 2022. A deep learning approach to detection and mitigation of distributed denial of service attacks in

Received: August 05, 2025

ISSN: 1311-1728 (printed version); ISSN: 1314-8060 (on-line version)

high availability intelligent transport systems. Mobile Networks and Applications, 27(4), pp.1423-1443.

13. Palekar, S. and Radhika, Y., 2022. IoT authentication model with optimized deep Q network for attack detection and mitigation. International Journal of Intelligent Robotics and Applications, 6(2), pp.350-364.
14. Popoola, S.I., Adebisi, B., Ande, R., Hammoudeh, M. and Atayero, A.A., 2021. Memory-efficient deep learning for botnet attack detection in IoT networks. Electronics, 10(9), p.1104.
15. Popoola, S.I., Adebisi, B., Hammoudeh, M., Gui, G. and Gacanin, H., 2020. Hybrid deep learning for botnet attack detection in the internet-of-things networks. IEEE Internet of Things Journal, 8(6), pp.4944-4956.
16. Qiu, H., Dong, T., Zhang, T., Lu, J., Memmi, G. and Qiu, M., 2020. Adversarial attacks against network intrusion detection in IoT systems. IEEE Internet of Things Journal, 8(13), pp.10327-10335.
17. Riaz, S., Latif, S., Usman, S.M., Ullah, S.S., Algarni, A.D., Yasin, A., Anwar, A., Elmannai, H. and Hussain, S., 2022. Malware detection in Internet of Things (IoT) devices using deep learning. Sensors, 22(23), p.9305
18. Sagu, A., Gill, N.S., Gulia, P., Priyadarshini, I. and Chatterjee, J.M., 2024. Hybrid Optimization Algorithm for Detection of Security Attacks in IoT-Enabled Cyber-Physical Systems. IEEE Transactions on Big Data.
19. Sahu, A.K., Sharma, S., Tanveer, M. and Raja, R., 2021. Internet of Things attack detection using hybrid Deep Learning Model. Computer Communications, 176, pp.146-154.
20. SAMPADA BHOSALE, (2018), https://www.kaggle.com/datasets/sampadab17/network-intrusion-detection/data.
21. Samy, A., Yu, H. and Zhang, H., 2020. Fog-based attack detection framework for internet of things using deep learning. Ieee Access, 8, pp.74571-74585.
22. Shobana, M., Shanmuganathan, C., Challa, N.P. and Ramya, S., 2022. An optimized hybrid deep neural network architecture for intrusion detection in real-time IoT networks. Transactions on Emerging Telecommunications Technologies, 33(12), p.e4609.
23. Taher, F., Abdel-Salam, M., Elhoseny, M. and El-Hasnony, I.M., 2023. Reliable machine learning model for IIoT botnet detection. IEEE Access, 11, pp.49319-49336.
24. Vu, L., Nguyen, Q.U., Nguyen, D.N., Hoang, D.T. and Dutkiewicz, E., 2020. Deep transfer learning for IoT attack detection. IEEE Access, 8, pp.107335-107344.
25. Wardhani, R.W., Putranto, D.S.C., Jo, U. and Kim, H., 2023. Toward Enhanced Attack Detection and Explanation in Intrusion Detection System-Based IoT Environment Data. IEEE Access, 11, pp.131661-131676.

Received: August 05, 2025