SSRG International Journal of Electrical and Electronics Engineering
ISSN: 2348-8379/ https://doi.org/10.14445/23488379/IJEEE-V12I7P116

Volume 12 Issue 7, 221-237, July 2025
© 2025 Seventh Sense Research Group®

Original Article

An Effective Big Data-Driven IoT Intrusion Detection Model using BAOA-IESNN

S. Ravishankar[1], P. Kanmani[2]

[1]*Department of Computer Science, Sona College of Arts and Science, Salem, Tamil Nadu, India.*
[2]*Department of Computer Science, Thiruvalluvar Government Arts College, Rasipuram, Tamil Nadu, India.*

[1]*Corresponding Author : ravirohan83@gmail.com*

Received: 11 May 2025 Revised: 13 June 2025 Accepted: 12 July 2025 Published: 31 July 2025

Abstract *- The rapid development of technological advances has saturated communications with network data traffic. Considering the multitude of sensor nodes within the Internet of Things (IoT) network, the analysis of network traffic data using conventional approaches could be difficult. It is essential to analyze this network traffic within a Big Data (BD) environment. BD refers to an enormous volume of complex data essential for analyzing patterns in networks and comprehending past occurrences within the network. Hence, this research proposes a BD-based intrusion detection model using the Deep Learning (DL) algorithm with Apache Spark (APS) for attack detection and classification. The developed intrusion detection model includes multiple processes such as data collection, data preprocessing, feature selection and classification. For this research, the BoT-IoT dataset is collected and applied to train and evaluate the model. The collected dataset is processed in the preprocessing stage with multiple preprocessing methods like data cleaning, label encoding, oversampling, and min-max normalization. The Binary Archimedes Optimization Algorithm (BAOA) technique is applied to select the optimal features from the dataset. Based on the selected optimal features, the Improved Elman Spiking Neural Network (IESNN) model performs attack detection and classification. The BAOA-IESNN model attained 99.39% accuracy, 99.24% detection rate, 99.30% precision, and 99.26% f1-score in binary classification, and 98.85% accuracy, 98.79% detection rate, 98.87% precision, and 98.83% f1-score in binary classification. This BAOA-IESNN model outperformed the current models compared in this research by demonstrating an effective performance in detecting and classifying attacks.*

Keywords *- Big Data, IoT, Intrusion detection, Deep Learning, BAOA, IESNN, Apache spark, BoT-IoT.*

1. Introduction

In recent years, the proliferation of data from computer and network systems, along with increasing security risks, has contributed to increasing the demand for intrusion detection. The necessity to identify errors and cyberattacks has directed research towards the automatic intrusion detection and classification in big datasets, where manual labeling is unfeasible due to the large volume of data. The outcomes derived from analysis of data can be utilized to produce alerts that predict irregularities to prevent system malfunctions and attacks [1]. The concept of IDS is the precise detection of diverse threats or attacks that can damage or disrupt an information system. An IDS could be classified into host-based, network-based, or a hybrid model. The host-based IDS concentrates mainly on the internal surveillance of a computer system. A host-based IDS performs duties such as file integrity verification, log analysis, and Windows registry monitoring. The network-based IDS monitors and evaluates network traffic to identify threats such as password attacks, SQL injection attacks, and Denial-of-Service (DoS) attacks. The quick expansion of computing applications and networks, such as IoT, has led to a rise in cyberattacks globally. The IDS can be classified as either signature-based or anomaly-based. A signature-based IDS comprises patterns for recognized attacks and is incapable of identifying unexpected threats. The signature-based IDS's database has to be regularly updated to remain current with all recognized attack signatures. On the other hand, an anomaly-based IDS detects variations from regular traffic patterns. Considering that numerous Machine Learning (ML) and DL methodologies can effectively be utilized for anomaly detection, it is logical to conclude that anomaly-based IDS represents a promising domain of research [2]. Intrusion detection has evolved into a BD challenge, characterized by a semantic imbalance between extensive security sources of data and real-time information regarding attacks. Intrusion detection is confronted with a significant BD challenge, regardless of whether it involves a singular source of data or a comprehensive architecture that consolidates data from diverse sources. To address this issue, the research model must implement BD handlers, specifically components that manage such data. Figure 1 shows the architecture of BD in handling data sources. Moreover, as managing such BD quantities beyond human capacity, employing ML/DL algorithms to get valuable insights from massive, multidimensional datasets appears to be the sole solution [3]. The term BD refers to a substantial volume of data characterized by a complex structure and intricate interrelations among the datasets. The principal advantage of BD is its superior capacity to analyze extensive datasets compared to conventional analytical methods. This reflects the growing interest in BD among the present generation, driven by advancements in data collection, storage, and analysis. The utilization of digital media has expanded across numerous sectors in recent years, producing vast quantities of data, including medical, financial, and social networking information. The expense of storing data is decreasing regularly, allowing for the ongoing storage of all data rather than its removal. Numerous techniques for analyzing data have been developed; however, just a few of them have been effectively applied to analyzed data. BD represents the aggregation of vast resources that could be utilized regularly [4].

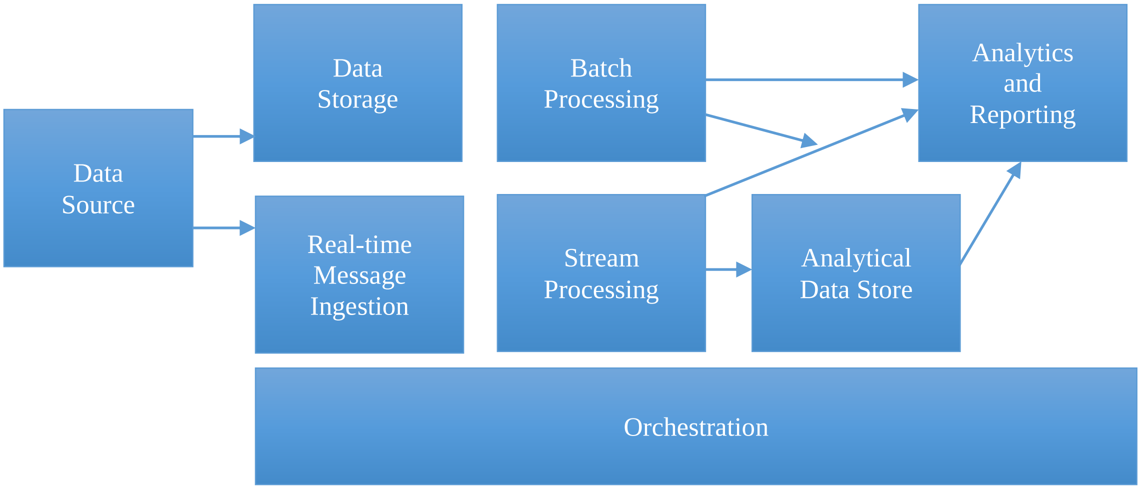

Fig. 1 Big data architecture

BD is the convergence of numerous technical advancements designed to enhance the capabilities of processing and the system's storage capacity.

Moreover, the substantial quantity of heterogeneous data is evaluated via BD analytics, which incorporates many modern and parallel data processing methodologies. Fog nodes aggregate extensive data for analysis and processing, encompassing images, text, and videos.

Nonetheless, data analysis methodologies provide numerous risks concerning the security and privacy of personal information, as data is acquired, stored, and analyzed from various internet sources. Consequently, Internet users are susceptible to privacy violations, as their private data may be utilized in various forms for diverse analyses, with these tools capable of obtaining sensitive and significant data. To address security concerns, various conventional methods for data protection are being utilized. Nonetheless, they cannot address these issues owing to the intricacy and diversity of BD [5].

As shown in Figure 2, the BD is based upon the 10 V's: Velocity, Variety, Veracity, Value, Variability, Visualization, Validity, Volatility, Vulnerability, and Volume. Additionally, data can be categorized into unstructured, semi-structured, structured, and structured types [6]. These ten Vs are described in Table 1 [7].

Table 1. Ten Vs of Big Data

V-Term	Description
Volume	Refers to the substantial quantity of data produced from different sources.
Velocity	The data speed when generated, analyzed, and processed.
Variety	The diversity of data formats (semi-structured, unstructured, and structured).
Veracity	The data precision, quality, and reliability.
Value	The benefits and insights that can be obtained from data analytics.
Variability	The inconsistency and unpredictability in data flows and formats.
Visualization	The ability to effectively present data insights through visual tools.
Validity	Ensures that the data is correct, meaningful, and relevant.
Volatility	Describes how long data is valid and should be stored or retained.
Vulnerability	The potential risks to data privacy and security during processing and storage.

The BD cybersecurity analytics focuses on improving cybersecurity operations by utilizing advanced analytical methods and BD solutions. Implementing BD analytics is rational for numerous reasons, including the innovative methodologies it provides to improve data analytics. Technologies employed in security that lack complete efficacy and are incapable of detecting cybersecurity, fraud, or threats breaches. Numerous BD analytics systems and technologies are presently accessible for detecting patterns in vast datasets as abnormal activities. The identification of relationships within large datasets is facilitated by a diverse array of ML/DL methodologies, including outlier detection, data visualization, and clustering [8].

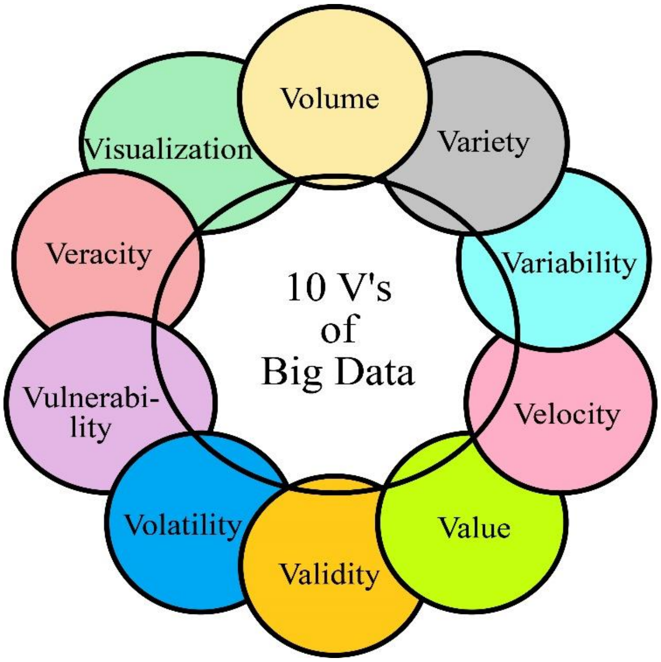

Fig. 2 Ten Vs of Big Data

1.1. Problem Statement

Cyberattack detection and prevention significantly depend on BD analytics, which assists organizations in collecting, analyzing, and extracting insights from extensive datasets generated from several origins. BD analytics algorithms can examine user activity, system logs, and network traffic data to identify anomalies that may indicate a cyberattack. Security issues can be identified more quickly if organizations establish standard behavioral patterns and observe deviations from these typical behaviors. BD analytics is employed to identify various cyberattacks, including phishing attacks, malware infections, and DDoS and DoS attacks [9]. Due to the rapid growth of IoT devices, there is an increased risk of cyberattacks worldwide. Implementing BD analytics with advanced models like DL enables efficient, accurate, and scalable intrusion detection to secure IoT environments in real time. Thus, a DL-based IDS model is developed to detect and classify attacks from the BD data set using the APS platform.

1.2. Research Objectives

The rapid development of IoT has presented different problems in data management. One of the main problems is particularly within the field of IDS that employs advanced ML and DL algorithms. These algorithms rely significantly on meticulously managed and extensive datasets to function optimally. The implementation of cybersecurity is necessary in developing resilient IDS models. Feature selection is crucial for optimizing IDS data sets. Effective feature selection techniques have demonstrated the capacity to reduce model training durations and augment the ability to generalize the IDS [10]. The key contributions of this proposed work are defined in the following:

- This research develops a big data-based intrusion detection model using a hybrid DL model for detecting and classifying attacks.
- A big data dataset called BoT-IoT is applied to train and evaluate the developed model.
- Multiple data preprocessing processes are performed to enhance the data's suitability for model training and analysis.
- The BAOA technique is applied for selecting the optimal features, and the IESNN algorithm is utilized to detect and classify the attacks.
- The developed BAOA-IESNN model is processed using Apache Spark on Google Colab.
- The research model is evaluated using performance indicators like accuracy, detection rate, precision, and F1-score.
- The results of the developed BAOA-IESNN model are compared and validated with the current models discussed in this research for proper validation.

The paper is organized into the subsequent sections. Section II succinctly examines the current models relevant to the research study. Section III encompasses the implementation of the developed research methodology. Section IV highlights the experimentation findings of the research methodology and a comparison with current models. The final section concludes the research with an overview of the findings and recommendations for subsequent research initiatives.

2. Related Works

In this section, a review of current works applied to improving big data-based intrusion detection has been conducted. All the reviewed current models are critically analyzed and presented in Table 2 with their advantages and limitations. The review of the related works is as follows: A two-stage IDS methodology utilizing the CSE-CIC-IDS2018 dataset was developed in [11]. The research utilized the APS-based method and Stacked Auto Encoder (SAE). The SAE methodology utilized an AE to acquire data-driven, non-linear representations of features. The APS methodology employed Principal Components Analysis (PCA) for dimensionality reduction. In these methodologies, the binary classification model initially distinguished between normal and abnormal traffic, producing probability results that were utilized as attributes in conjunction with a restricted set of features for training a multi-class classification model for predicting attacks. The SAE methodology consistently surpassed ASpark in results, but the ASpark methodology excelled in computational efficiency.

A BD framework to identify intrusions in smart grid systems was proposed in [12] utilizing AdaBelief Exponential Features Selection with Kernel-based Extreme Neural Networks (AEFS-KENN). The AEFS was employed to effectively manage substantial datasets from the smart grid to enhance security. The KENN approach was employed to efficiently predict security problems. The Polar Bear Optimizer (PBO) approach was employed to effectively ascertain the parameters for estimating the radial basis function. The findings indicated that the framework enhanced attack detection efficacy. The research in [13] presented three models: Apache Spark, Long Short-Term Memory (LSTM), and a Convolutional Neural Network (CNN) for enhancing the detection of various attack types. Random Forest model (RF) was utilized to choose the significant features for dimensionality reduction. Undersampling and Oversampling techniques were utilized to mitigate the data imbalance ratio. The Apache Spark model yielded superior outcomes in comparison to the CNN and LSTM models.

A security-based data analysis utilizing the Apache Spark BD analytics engine was implemented in [14]. A prototype IDS was created to identify data abnormalities with ML, specifically employing the k-means technique for clustering analysis incorporated into Spark's MLlib. The feature extraction for anomaly detection was presently difficult due to the lack of active and thorough monitoring of the anomaly detection problem. To improve the efficacy of ML-based IDS, a Big Data-based Hierarchical DL System (BDHDLS) was developed in [15]. BDHDLS utilized content and behavioral features to analyze network traffic features and the data included in the payloads. All DL models, such as CNN, RNN, and CNN-RNN in the BDHDLS, focused on learning the distinct data distribution within a single cluster. BDHDLS enhanced classification performance relative to the other three models.

An IDS model utilizing anomaly detection with the LSTM algorithm and the NSL-KDD dataset was developed in [16]. The model was trained on normal behavior for the identification of both known and unknown attacks. The results indicated that employing BD approaches and DL algorithms enhanced detection accuracy while decreasing false alarm rates, a critical factor in anomaly-based IDS. EffiGRU-GhostNet, a DL-based ensemble framework, was designed in [17] for high-accuracy detection of DDoS while minimizing resource utilization. The framework integrated EfficientNet-Gated Recurrent Units (GRU) with the GhostNet framework, refined via Principal Component Analysis with Locally Preserving Projection (PCA-LLP) to proficiently manage extensive datasets. The results suggested that the framework was a dependable, scalable approach for detecting DDoS, expanding the domain of big data-based security. The research in [18] addressed the intricate challenges associated with handling large data quantities and uneven class distributions in detecting intrusions, resulting from the rapid increase in network traffic.

A hybrid methodology was presented to enhance the precision of detecting minority groups in imbalanced datasets. The hybrid methodology integrated K-means clustering with the RF classifier, designed for BD processing using Hadoop and PySpark. The research attained optimal efficacy in detecting intrusions. The study in [19] examined the drawbacks of existing IDSs by analyzing and assessing ten ML and DL models through feature selection techniques. The experiment was conducted on a consolidated complex data set comprising dual IoT and local traffic data sets. The ranking and best selections methodology (RBSM) was applied to evaluate the performance of every measure for every model, facilitating the development of a reliable, flexible, and scalable ensemble classification system that automatically adapts to the integration of ML/DL models or in the IoT network. The findings indicated that the ensemble model attained the maximum accuracy in performance.

An optimization-based DL method for IDS with the Spark model was proposed in [20]. The fuzzy local information and Bhattacharya-Based C-Mean (FLIBCM) was developed by integrating Bhattacharya distances with FLI-C-Means (FLICM). IDS was accomplished utilizing the Deep Maxout Networks (DMN), applied with the Students' Psychology Water Cycles Caviar (SPWCC) method. This model demonstrated superior performance with the best accuracy. The Two-phase-IDS (TP-IDS) in [21] was developed in dual steps to enhance performance. Step I of the model employed k-Nearest Neighbors (kNN) and Support Vector Machines (SVM) algorithms. In step II, Naïve Bayesian (NB) and Decision Trees (DT) were employed, serving as the validation step of the system to enhance accuracy. Both stages utilized the Hadoop Distributed File Systems (HDFS) as the fundamental storage and processing platform, which enabled parallel processing to enhance system speed and thus improve results.

An IDS model using the Spark-Chi-SVM method for detecting intrusions on the BD environment was proposed in [22]. The model employed ChiSqSelector for feature selection and constructed an IDS model utilizing a SVM classifier on the APS BD framework. The experimental results indicated that the Spark-Chi-SVM model exhibited superior performance, minimized training duration, and was effective for BD applications. A hybrid DL method that integrated CNN and LSTM models was developed in [23] for the detection of DDoS attacks. The research utilized PySpark with Apache

Spark within the Google Colaboratory platform. The dataset's features were diminished by the correlation approach, guaranteeing the retention of the best features in the analyses. The DL method, incorporating one-dimensional CNN and LSTM, achieved the maximum accuracy. An effective classification technique for IDS, comprising two algorithms, utilizing the CSE-CIC-IDS-2018 data set, was proposed in [24]. The research investigated the utilization of RF for selecting features along with ML methodologies, including Linear Regression (LR), k-NN, Bayesian methods, Classification And Regression Trees (CART), RF, Extreme Gradient Boosting (XGBoost), and Multi-Layer Perceptron (MLP), to implement IDS. The outcomes of the experiment indicated that the MLP method exhibited the highest performance. An ML-based network IDS model that employed Random Oversampling to mitigate data imbalance was developed in [25], which utilized Stacking Feature Embedding derived from clustering outcomes, and incorporated PCA for dimensionality reduction, designed specifically for imbalanced and big datasets. The DT, RF, Extra Tree (ET), and XGB models were assessed for binary and multiclass classification, with the RF model demonstrating consistent high performance across several datasets. An approach for selecting beneficial features in network intrusions utilizing fuzzy numbers and scoring techniques derived from Correlation Feature Selection (CFS) for IDS was developed in [26]. The technique defined the number of features as a fuzzy number for eliminating inefficient features and minimized data dimensions, with the heuristic technique of the correlation-based feature selection algorithm represented as a triangle fuzzy number membership function. The findings indicated that the method identified fewer characteristics than traditional methods while achieving a better detection rate.

Table 2. Critical analysis of analyzed related works

Ref.	Models	Application	Dataset	Advantages	Disadvantages
[11]	SAE & Apache Spark with PCA	Multi-stage IDS	CSE-CIC-IDS2018	SAE showed superior accuracy; ASpark had high efficiency.	Limited by fixed feature sets and not generalized well.
[12]	AEFS-KENN with PBO	Smart grid intrusion detection	SGCC, CICIDS 2017, ICS, and UNSW-NB 15.	Handles large datasets; improved parameter estimation.	High computational complexity.
[13]	Apache Spark, CNN, LSTM with RF	Attack detection across models	CSE-CIC-IDS2018	Spark achieved better results and handled the imbalance.	Oversampling caused overfitting.
[14]	Apache Spark with k-means	Real-time anomaly detection	Custom Dataset	Scalable prototype using Spark.	Limited feature extraction capability.
[15]	BDHDLS (CNN, RNN, CNN-RNN)	DL-based big data IDS	DARPA1998, ISCX2012, and CICIDS2017.	Improved performance via hierarchical features.	The model has not generalized across all datasets.
[16]	LSTM on NSL-KDD	Anomaly-based IDS	NSL-KDD	Accurate with fewer false positives.	The dataset is outdated; it lacks IoT traffic.
[17]	EffiGRU-GhostNet with PCA-LLP	DDoS detection	APADDoS and IoT-23	Efficient with low resource use.	Models struggle with unseen attack types.
[18]	K-means + RF on Hadoop & PySpark	Big data intrusion detection	NSL-KDD	High detection for minority classes.	A complex hybrid approach reduced interpretability.
[19]	Ensemble ML/DL with RBSM	IoT and local data	ToN_IoT, Aposemat IoT-23, and UNSW-NB15.	Adaptive and high-performing.	Heavy reliance on feature selection accuracy.
[20]	FLIBCM + DMN with SPWCC	Spark-based big data IDS	NSL-KDD	High accuracy using fuzzy and Maxout models.	Model tuning is computationally expensive.
[21]	TP-IDS (SVM, kNN, DT, NB)	HDFS-based IDS	NSL-KDD	Enhanced speed via parallel processing.	Two-phase validation adds complexity.
[22]	Spark-Chi-SVM	Spark-based IDS	KDD99	Fast training; suited for big data.	The model did not perform well in multi-class settings.
[23]	CNN-LSTM	DDoS detection	CICIoT2023 and	Best accuracy with	The model did not scale

	hybrid in PySpark		TON_IOT	reduced feature space.	well across different attack types.
[24]	RF feature selection + various ML models	IDS using CSE-CIC-IDS-2018	CSE-CIC-IDS-2018	MLP outperformed others.	Evaluation limited to a specific dataset.
[25]	Stacking Feature Embedding + PCA	Binary and multiclass IDS	UNSW-NB15, CIC-IDS-2017, and CIC-IDS-2018	Handled imbalance well; high RF performance.	Risk of model complexity and overfitting.
[26]	Fuzzy scoring + correlation-based FS	Network intrusion detection	NSL-KDD and CIC-IDS-2017	Select fewer, more relevant features.	Needs fine-tuned membership functions.

2.1. Research Gap Analysis

From reviewing the current works on BD-based intrusion detection models, there exist some research gaps in the analyzed current models. All these current models lack comprehensive scalability in IoT network data and are not generalized across various and imbalanced datasets. The majority of the models have not utilized feature selection with classification. And also, some models utilized a few outdated datasets such as NSL-KDD and KDD99. These datasets are limited to modern attack types. Some models utilized hybrid models that caused increased computational complexity and reduced the performance.

Few research works employed both feature selection and advanced DL models combined with the Apache Spark BD tool. Those models highlight better performance and results for IoT networks. Hence, this research aimed to address these gaps by integrating optimized feature selection and a DL model on Apache Spark.

3. Materials and Methods

This research proposed an IoT intrusion detection and classification model based on BD using a metaheuristic and DL algorithm model. The model is developed and deployed on the Apache Spark BD platform for processing and classification.

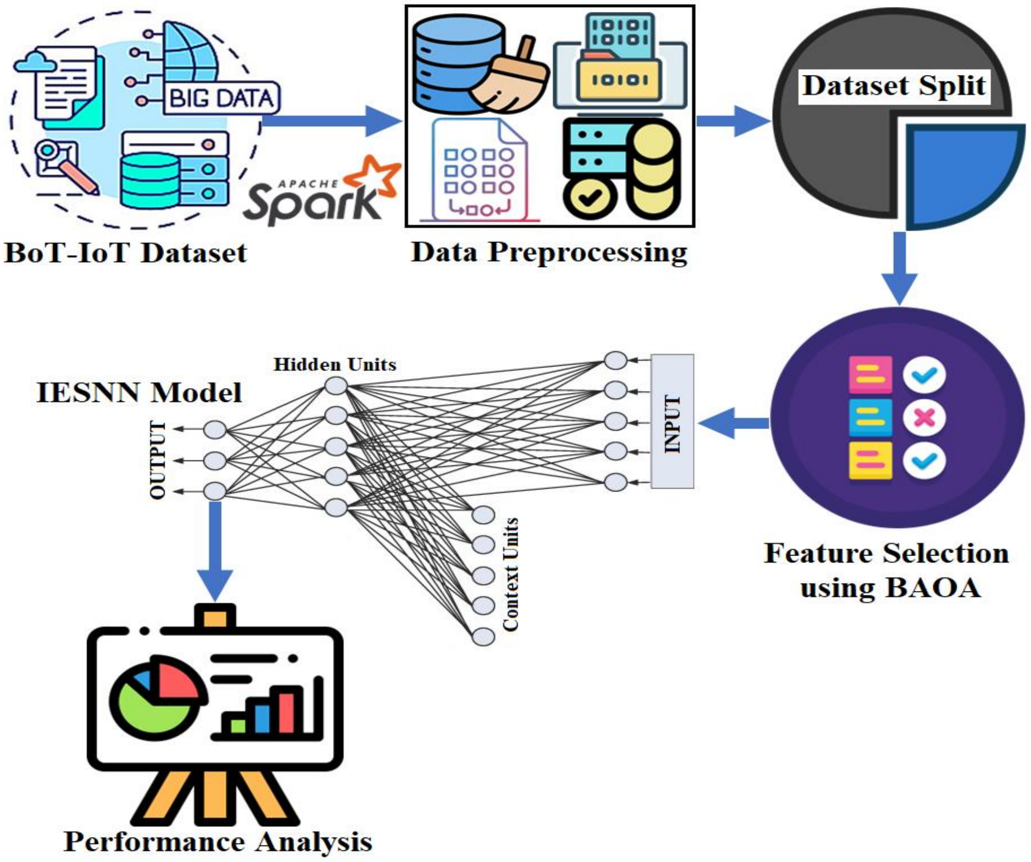

Fig. 3 Proposed BAOA-IESNN-based Big Data IDS model

Apache Spark will be helpful in handling the BD dataset, like the BoT-IoT dataset, in processing, attack detection, and classification. Figure 3 depicts the workflow of the developed BAOA-IESNN intrusion detection model using BD. The developed intrusion detection model has multiple primary stages of processes such as data collection, data preprocessing, feature selection and classification. Initially, the BoT-IoT dataset is collected and applied to the research. This BoT-IoT dataset is classified as a BD dataset, as it highlights all the characteristics of BD. The collected dataset is processed in the preprocessing stage with multiple preprocessing methods like data cleaning, label encoding, random oversampling, and normalization. After preprocessing, the dataset is split into an 80:20 ratio for training and evaluating the model. The majority of the data set was applied to train the model, and the remaining data was employed for evaluation.

The preprocessed training set of the data is then processed to choose the optimal features from the data set using BAOA. These optimal features are helpful in improving the classification performance. Based on the selected optimal features, the IESNN model performs attack detection and classification. The IESNN model is effective in capturing temporal dependencies and dynamic patterns in the data.

This is due to the model's spiking mechanisms integrated with feedback memory. Finally, the performance of the BAOA-IESNN model is evaluated based on performance indicators like accuracy, detection rate, precision, and F1-score. The results will be evaluated and compared with the current models for validation. The results will highlight that the developed BAOA-IESNN model is highly accurate and effective in detecting intrusions on both binary and multiclass classifications.

3.1. Dataset Details

The Bot-IoT dataset was developed by the Cyber Range Laboratory at UNSW Canberra within an actual network setting.

Table 3. Dataset distribution

Attack Classes	No. of Records	Category
Benign	9543	Normal
Data Theft	118	Information Theft
Keylogging	1469	
DoS-HTTP	29706	DoS
DoS-TCP	12315997	
DoS-UDP	20659491	
DDoS-HTTP	19771	DDoS
DDoS-UDP	18965106	
DDoS-TCP	19547603	
OS Fingerprinting	358275	Information Gathering
Service Scanning	1463364	

This dataset comprised data collected from diverse smart home equipment, including refrigerators, lighting systems, thermostats, garage doors, and weather monitoring devices. This network architecture comprises a combination of botnet and benign traffic covering 73 million records. The Bot-IoT dataset comprises 43 network traffic attributes and three classifications of labels. Nevertheless, only 37 of the 43 features were considered significant for detecting botnet attacks in IoT networks. Table 3 presents the BoT-IoT dataset details. The two categories of attacks, specifically keylogging and data theft, have an insufficient number of data samples. It can adversely affect the classification outcomes. Consequently, these two attack classes are excluded from the experiment [27].

3.2. Apache Spark

Apache Spark is a cluster computing framework designed primarily for the processing of vast quantities of data. This is an open-source software. It employs a multistage memory-based processing method that achieves processing speeds 100 times faster than map-reduce techniques. Spark can handle Hadoop clusters, access and analyze any Hadoop data source, and execute Hadoop tasks. Numerous initiatives have been recently undertaken to develop a comprehensive learning system in Apache Spark. On December 30, 2016, the Intel research team introduced BigDL, a DL library designed for distributed use with Apache Spark, using CPU resources and built on parallel data models and functional computing, along with support for emerging technologies. This incorporates DL with Apache Spark to offer a unified DL computing framework for data processing. Figure 4 illustrates the relation of BigDL with Apache Spark.

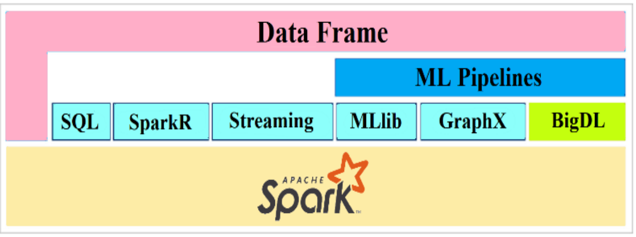

Fig. 4 BigDL in apache spark

To construct the developed research, the BD analytical tool was utilized. This technology is primarily categorized into platforms and software derived from the Hadoop ecosystem. The Hadoop ecosystem comprises various platforms and components that work together to yield the result. This implementation utilized Hadoop 2.8.0 for data storage and distribution. Additionally, Spark 2.2.0 serves as a data processing platform. An advantageous aspect of implementing the model was the utilization of the BigDL library. Spark has multiple libraries, and the BigDL library was utilized for this research due to its implementation of DL algorithms. Initially, the data records are retrieved from the dataset and transferred from Hadoop to HDFS. Subsequently, the data is converted to

RDD; at this stage, the research model is constructed within the BigDL library. The model and data are processed for distribution across the cluster among each node for distributed training using the Spark and BigDL library [28].

3.3. Preprocessing

It is imperative to address the issues of missing data, duplicate entries, and data on characters within the dataset to alleviate their negative effects and enhance the overall data quality. This can be accomplished by pre-processing approaches to minimize noise and abnormalities, hence enhancing the data's suitability for training and analysis with models based on DL. The subsequent steps were executed to achieve this:

Data cleaning is an essential preprocessing phase that guarantees the quality and dependability of the dataset utilized for developing an IDS in IoT scenarios. This method encompasses multiple technical processes designed to refine raw data, hence improving the efficiency and precision of the IDS. The essential steps comprise:

Elimination of duplicate rows: Every record in the data set has to be distinct to prevent duplicated information, which may unbalance the model.

Fixing missing values: Mitigating biased analysis and model projections necessitates the resolution of incomplete data.

Outlier identification and exclusion: Detecting and handling data points that dramatically differ from the rest of the dataset is essential, as these could represent noise or infrequent occurrences.

To properly implement the selection of features and classification model, it is important to transform categorical attributes into numerical attributes. The procedure of feature encoding guarantees that categorical attributes are expressed by their respective numerical values. To perform this, the Label Encoder is employed, assigning a distinct integer for every categorical attribute according to their alphabetical sequence. Transforming categorical attributes into numerical values facilitates the effective processing of data by selecting features and a classification model.

Feature resampling is a technique employed to rectify the imbalance among features within a dataset. Random Oversampling (RO) provides a simplistic approach to rebalance class distribution in an imbalanced dataset. It generates random replication of cases from the minimal category and integrates them into the set of training data. i.e., if the dataset’s class ratio is 80:20, then 20 indicates the minimal class and 80 indicates the maximal class. It is effective for algorithms with uneven distributions and for a class capable of adapting to the model by duplicating instances. This methodology addresses a significantly unbalanced dataset, where resampling optimization is essential for balancing the dataset to enhance performance while preventing overfitting [29].

The data normalization or standardization approach transforms the complete value range of a feature set into a specified range. The variability in data values across features can significantly impede the DL system's training process. For instance, a particular feature may possess a value range of [0, 1000], whilst another may be limited to a range of [0, 1]. In this scenario, without standardization, a feature with a broader range of value will have greater influence in the algorithm's training. Min-max normalization was employed to transform all data into the interval [0, 1], hence accelerating model convergence and enhancing classification accuracy, as given in Equation (1):

$$X' = \frac{X-Min}{Max-Min} \tag{1}$$

Here, Max denotes the feature’s maximum value, Min signifies the feature’s minimum value, X represents the original value of the feature, and X' represents the normalized value of feature [30].

3.4. Binary AOA-based Feature Selection

The AOA is a metaheuristic technique that examines the principle wherein an object submerged in a fluid experiences a buoyant force that is upward equivalent to the fluid's weight it displaces. It also indicates that the buoyant force acting on an object in a fluid is correlated with the object's physical volume, density, and acceleration while in motion. When a specific point is attained, the net force acting on the thing within the fluid is null, resulting in the body in a state of equilibrium. The definitive mathematical representation of the AOA is formulated as follows [31]:

Initially, a state transition function TF is introduced to characterize the shift between global and local searches of an object within a fluid.

$$TF = \exp\left(\frac{t-t_{max}}{t_{max}}\right) \tag{2}$$

Here in Equation (2), t denotes the current iteration count, whereas t_{max} signifies the maximum iteration limit. In this context, $TF = 1$ signifies that the object has attained a state of equilibrium.

In addition, the density factor ‘d’ for the subsequent iteration ‘t+1’ is introduced to more accurately characterize the transition of items between global and local searches. It is described as expressed in Equation (3):

$$d^{t+1} = \exp\left(\frac{t_{max}-t}{t_{max}}\right) - \frac{t}{t_{max}} \tag{3}$$

If $d = 0$, an object is in a state of equilibrium. In the end, the acceleration factor acc_i^{t+1} for an object *i, it* is delineated as in Equation (4) to characterize the impact across physical objects:

$$acc_i^{t+1} = \begin{cases} \frac{dens_{mr} + vol_{mr} \times acc_{mr}}{dens_i^{t+1} \times vol_i^{t+1}} & TF \leq 0.5 \\ \frac{dens_{best} + vol_{best} \times acc_{best}}{dens_i^{t+1} \times vol_i^{t+1}} & TF > 0.5 \end{cases} \tag{4}$$

Here, vol_i^{t+1}, $dens_i^{t+1}$, and acc_i^{t+1} denote the acceleration, density, and volume of the object at time t, correspondingly. vol_{mr}, $dens_{mr}$, and acc_{mr} denote the volume, density, and acceleration of the random object, correspondingly; $dens_{best}$, vol_{best}, and acc_{best} denote the acceleration, density, and volume of the ideal object, respectively.

The accelerating factor acc varies in two phases: if $TF \leq 0.5$, a collision occurs between objects, and the update of acceleration is dependent upon the determining factor of random objects. When TF exceeds 0.5, no collision occurs between objects, and the acceleration update is dependent upon the factors determining the current optimal object. The associated acceleration parameter is normalized using the following Equation (5):

$$acc_{i-norm}^{t+1} = u \times \frac{acc_i^{t+1} - \min(acc)}{\max(acc) - \min(acc)} + k \tag{5}$$

Here, k and u represent constants. Throughout the entire operation, the object alters its position in accordance with two stages, dependent upon the variation of acceleration (Equation (6):

$$x_i^{t+1} = \begin{cases} x_i^t + c_1 \times rand \times acc_{i-norm}^{t+1} \times d \times (x_{rand} - x_i^t) \; TF \leq 0.5 \\ \begin{cases} x_{best}^t + c_2 \times rand \times acc_{i-norm}^{t+1} \times d \times (T \times x_{best} - x_i^t) \; P < 0.5 \\ x_{best}^t - c_2 \times rand \times acc_{i-norm}^{t+1} \times d \times (T \times x_{best} - x_i^t) \; P \geq 0.5 \end{cases} TF > 0.5 \end{cases} \tag{6}$$

In this context, $rand$ denotes a uniformly distributed random number within the interval [0, 1], T is defined as c_3 multiplied by TF, and $P = 2 \times rand - c_4, c_1, c_2, c_3$ and c_4 were constants, while x_i^{t+1}, x_{rand}, and x_{best} signify the position of object *i* at time $t + 1$, the positioning of a randomly selected object, and the position of the optimal object, accordingly. The AOA possesses a robust capacity to equilibrate both local and global search capabilities, exhibiting minimal dependence on the quantity of adaptive parameters. As a population-based algorithm, AOA is subject to converging on local optimal solutions due to diminishing population variety in later iterations, potentially leading to slow algorithm convergence. AOA is intended to address continuous issues and is not applicable to discrete issues. This research employs a binary variant of AOA integrated with a V-shaped transfer function for feature selection. This binary variant significantly enhances exploration capabilities by converting the continuous search space into a binary space. A transfer function is required to convert the continuous search space into the discrete domain to adapt AOA for discrete issues. The function of transfer discretizes the method by assessing the values of various solutions across many dimensions. The V-shaped transfer function is a traditional transfer function that facilitates the conversion of an algorithm from continuous to discrete representation without altering the method's overall structure. The traditional representations of V-shaped transfer functions are illustrated in Equations (7) and (8):

$$t_1(x) = |\arctan(x)| \tag{7}$$

$$t_2(x) = \left| \frac{x}{\sqrt{1+x^2}} \right| \tag{8}$$

Here, x denotes the object's initial position. Given that the transfer function is employed to convert a continuous search space into a binary field, the variation in the function values of the transfer function must reside between 0 and 1. Due to the significant alterations in the conventional V-shaped transfer function within the interval [-5,5], which adversely impacts algorithm performance, this work introduces the t_3 function as given in the following Equations (9) to (11):

$$t_3(x) = \left| \arctan(x) \times \frac{x}{\sqrt{1+x^2}} \right| \tag{9}$$

The V-shaped function includes a scaling factor a.

$$t_3(x) = a \times \left| \arctan(x) \times \frac{x}{\sqrt{1+x^2}} \right| \tag{10}$$

To ensure the V-shaped function's value remains between 0 and 1, the limit must satisfy the following conditions:

$$\lim_{x \to \infty} a \times \left| \arctan(x) \times \frac{x}{\sqrt{1+x^2}} \right| \leq 1 \tag{11}$$

It is evident that once the threshold on the left-hand side of this inequality equals 1, the value of a is $\frac{2}{\pi}$. To fulfil the previously stated limitation, the scaling factor a must be below 0.64. The performance criteria for the function t_3 employed a distinct scaling factor a ($a \leq 0.64$). To assess both the local and global search capabilities of the algorithm utilizing the V-shaped transfer function t_3. The performance criteria are delineated as in Equation (12).

$$L = \frac{\sum_{i=1}^{n} x_i^t \oplus x_{best}}{n} \tag{12}$$

Here, L denotes the mean distance of every object to the global optimum value, $\oplus$ signifies the XOR operation between the current and optimal positions across every dimension, and n indicates the total number of objects. The greater the value of L, the more robust the global search capability; conversely, a smaller value of L enhances the local search capability. When $a = 0.64$, the mean value of the performance criteria L is maximized, indicating that at this value, the function t_3 exhibits a robust global search capability. The scaling coefficient a of the V-shaped transfer function t_3 employed in this work is 0.64.

The location updates of BAOA could be derived by integrating Equation (6) with Equation (10) in the following manner:

$$X_i^{t+1} = \begin{cases} 1 & rand < t_3(x_i^t) \\ 0 & rand \geq t_3(x_i^t) \end{cases} \tag{13}$$

Here in Equation (13), X_i^{t+1} denotes the position of object i at time $t + 1$. In contrast to the conventional AOA, the V-Shaped BAOA ignores the issue of algorithm performance being influenced by varying initial object positions. The Fitness Function (FF) aims to increase the classification accuracy of the IESNN model while reducing the number of features. The FF formula for the implemented BAOA technique is expressed in the following Equation (14).

$$FF = \alpha \cdot \left(1 - \delta(S)\right) + \beta \cdot \left(\frac{|S|}{D}\right) \tag{14}$$

Here, α represents the weight for the classification performance, which is set to 0.9. $\delta(S)$ represents the classification accuracy of the IESNN model using features in the selected feature subset S. β represents the weight for feature minimization, which is set to 0.1. $|S|$ represents the count of selected features, which is 12. D represents the total number of features in the dataset, which is 43 [32].

A total of 12 significant optimal features are selected from the BoT-IoT dataset using the BAOA approach. The selected optimal features were AR_P_Proto_P_SrcIP, Bytes, Dur, Dport, Daddr, Flgs_number, N_IN_Conn_P_DstIP, Proto_number, Seq, Sport, Saddr, and State_number.

3.5. IESNN-based Attack Classification Model

Following the feature selection technique, the selected features are input into the IESNN for classification. This partial recurrent SNN model variant enhances the IESNN by incorporating fundamental Elman Neural Network (ENN) principles. The topology of an ENN typically comprises the layers such as input, hidden, context and output. The context layer utilizes the positive-feedbacks mechanism to retain the prior outputs of the hidden layers, incorporating self-feedback with variable gain. The EESN's lightweight design enables it to be suitable for computationally intensive applications in embedded platforms and edge devices [34]. The layer of input and nodes is represented as shown in Equation (15).

$$y_i^{(1)} = f_i^{(1)}\left(net_i^{(1)}(m)\right); i = 1 \tag{15}$$

Let $net_i^{(1)}(m) = e_i^{(1)}(m) : n$, where n denotes the nth iteration, $e_i^{(1)}(m)$ serves as the input, and $y_i^{(1)}(m)$ represents the output of the first layer. The dynamics of IESNN are delineated in Equations (16) to (18).

$$NS(y) = NLF\left(W_{con*inv} NS_{con}(y), W_{in*inv} input(y)\right) \tag{16}$$

$$NS_{Conv}(y) = \alpha(y) NS_{con}(y-1) + W * NS(y-1) \tag{17}$$

$$output(y+1) = W_{inv*out} NS(y) \tag{18}$$

Here, $NLF(.)$ is a nonlinear variable that delineates the classification and presentation of IESNN. The output and input are indicated by the output (y), accordingly. In the context and hidden layers, the state node matrix denotes $NS(y)$ and $NSCon(y)$, accordingly. The variables W_{in*inv} and $W_{con*inv}$ denote neurons associated with weights of the data layer to the inferred layers, whereas $W_{inv*out}$ signifies neurons that weigh the various levels of invisible to output. The formula in Equation (19) denotes the node within the hidden layer.

$$y_j^{(2)}(n) = S\left(\left(net_j^{(2)}(n)\right); j = 1, \dots 9\right) \tag{19}$$

$$net_j^{(2)}(n) = \sum_i W_{in*inv} \times y_i^{(1)}(n) + \sum_k W_{con*inv} \times y_k^{(3)}(n); k = 1, \dots 9 \tag{20}$$

In the expression (20), $S\left(net_j^{(2)}(n)\right)$, a sigmoid function is denoted, where $y_i^{(1)}(n)$ and $y_k^{(3)}(n)$ signify the data from the hidden and input layers, respectively, and $y_j^{(2)}(n)$ denotes the output of the hidden layer. Subsequently, nodes within the level of context are represented by Equation (21) pertaining to the layer.

$$y_k^{(3)}(n) = \alpha y_k^{(3)}(n-1) + y_j^{(2)}(n-1) \tag{21}$$

Consider that α represents a self-connecting feedback gain that is updated within the context layer to achieve accurate intrusion detection and classification. Every layer-to-layer connection in the IESNN consists of a cluster with the same quantity of neural endpoints. The delay and weight are unique for every sub-connection. The single-layered IESNN comprises two neurons for input in the hidden layers and two output neurons. The outcome of the layer's nodes was illustrated at the output layer's end as in Equation (22).

$$y_l^{(4)}(n) = f_l^{(4)}\left(net_l^{(4)}(n)\right) \tag{22}$$

In this context, $f_l^{(4)}$ denotes the parameter regulated by the proposed IESNN methodology.

$$net_l^{(4)}(n) = \sum_j W_{inv*out} \times y_j^{(2)}(n) \tag{23}$$

Whereas in Equation (23), $W_{inv*out}$ represents the neural weights that connect the hidden layer outputs to the neurons. The network is altering the connection's metrics. The subsequent Equation (24) is employed to enhance the final phase of classification:

$$W_{inv*out}^{y}(Time+1) = W_{inv*out}^{y}(Time) - \eta \cdot \delta_f \cdot NS^{y} \tag{24}$$

Here, η denotes the learning rate, whereas $Time$ represents the unit for time. The threshold score of the neurons assesses the IDS classification for each neuron's spike during the specified time interval $Time$. This suggests that the threshold value of the considered neurons successfully differentiates intrusions from normalcy. Membrane potential η is classified into distinct attack types if it exceeds a predetermined threshold value. The subsequent Equation (25) can be utilized to calculate the delta function of the neurons, represented by the variable δ_f.

$$\delta_f = \frac{Error}{\sum_{i=1}^{Niv}\sum_{y=1}^{NoD} W_{inv*out}^{y}\frac{\partial input}{\partial Time}} \tag{25}$$

The following formula (26) computes the disparity between the ends of neurons.

$$Error = t_f^{DST} - Time_f^{NLF} \tag{26}$$

Here, t_f^{DST} denotes the length of time of a neuron's spike, whereas $Time_f^{NLF}$ represents the actual timing of a neuron's spike [35]. The pseudocode of the developed BAOA-IESNN model is presented below.

Algorithm: BAOA-IESNN Model

```
Initialization
Input: BoT-IoT dataset
Output: Classified labels (Benign / Attack types)
  Load BoT-IoT dataset into Apache Spark DataFrame
  Clean missing, null, or corrupted records
  Apply Label Encoding on categorical columns
  Perform Random Oversampling to balance class distribution
  Normalize features using Min-Max Normalization
Initialize Binary Archimedes Optimization Algorithm
  Set parameters: population size (N), max iterations (MaxIter), archimedes constant
  Initialize binary population P with random 0s and 1s (representing feature subset)
  Evaluate the fitness of each individual using the IESNN classifier accuracy on training data
While (t < MaxIter) do
  For each individual in population P:
    Calculate density, volume, and acceleration based on Archimedes' principle
    Update position (binary vector) using BAOA velocity and transfer function
  Evaluate the new fitness of updated individuals using IESNN
  Update the best solution if a better fitness is found
  t = t + 1
Select the optimal feature subset F_best from the best binary solution
Improved Elman Spike Neural Network Training
  Input: Dataset with selected features F_best
  Split data into training and testing sets (80:20)
  Initialize IESNN:
    Input, context, hidden, and output layers
    Spike encoding (e.g., Leaky Integrate-and-Fire model)
    Elman feedback from hidden to context units
  Train IESNN on training data
  Validate on test data
Perform both binary and multiclass classification
End
```

Table 4. Hyperparameter Values Set for the Research Model

Hyperparameter	Value
BAOA Population Size	50
No. of iterations of BAOA	1000
Transfer Function	Sigmoid
Acceleration	0.5
Archimedes constant	0.2
IESNN Hidden Layers	64
Context Neurons	64
Output Neurons (Binary/Multiclass)	2/9
Epoch	100
Batch Size	128
Activation Function	ReLU
Spike Encoder	Leaky Integrate and Fire
Optimizer	Adam

Table 4 shows the hyperparameter values set for the developed research model BAOA-IESNN.

The hyperparameter tuning ensures effective learning and model convergence while minimizing overfitting and optimizing computational efficiency for BD in Spark-based environments.

4. Experimentation Analysis

4.1. Experiment Setup

This section highlights the experiments conducted on the extensive BoT-IoT dataset using the developed BAOA-IESNN model. The study employed the PySpark tool, which facilitates programming with Python on the Apache Spark BD platform within the Google Colab environment. The proposed system is implemented using PySpark. It is a library that connects Python with Apache Spark. All testing was conducted on Windows 10 64-bit, utilizing a Core i7 processor operating at 2.70GHz, 16 GB of RAM, and the programming language, Python. The dataset is divided into testing and training halves of 20% and 80%, respectively.

4.2. Result Metrics

This research utilizes multiple performance indicators to assess the efficiency of the developed BAOA-IESNN intrusion detection model. The measurements encompass accuracy, Detection Rate (DR), precision, and F1-score.

TP-True Positive is the count of properly classified samples that are positive. TN-True Negative is the count of properly classified samples that were negative. FP-False Positive is the count of erroneously identified samples that were negative as positive. And FN-False Negative is the count of erroneously classified samples that were positive as negative. The performance metrics could be delineated as presented in the following Equations (27) to (30).

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{27}$$

$$Precision = \frac{TP}{TP+FP} \tag{28}$$

$$Detection\ Rate = \frac{TP}{TP+FN} \tag{29}$$

$$F1score = 2 \times \frac{Recall \times Precision}{Recall+Precision} \tag{30}$$

Accuracy evaluates the model's capacity to accurately categorize outcomes, providing a comprehensive performance assessment. Precision assesses the dependability of positive prediction and aims to reduce the false positives. This metric is especially important in IDS to prevent redundant false alarms, which could degrade the effectiveness of the system. The detection rate, or recall, evaluates the model's ability to identify true intrusions. The DR with higher recall values is necessary to minimize missed detections and provide a resilient system efficient at thoroughly detecting attacks. The F1-score is a harmonic average of recall and precision. It represents a balanced assessment of the classifier’s total efficacy, which focuses on its capacity to uphold efficiency and accuracy. A high F1-score signifies the model's efficiency in attaining an ideal balance between recall and precision [11-26].

4.3. Performance Evaluation

This section discusses the results of the developed research model BAOA-IESNN evaluated using the accuracy, DR, precision, and recall. Based on this BD framework, the results of the developed model are assessed in terms of both binary classification and multiclass classification. The results of the multiclass classification are compared with the current models for proper validation.

Table 5. Binary classification results of the research model

Metric (%)	Training	Testing
Accuracy	99.74	99.39
Detection Rate	99.59	99.24
Precision	99.71	99.30
F1-score	99.58	99.26

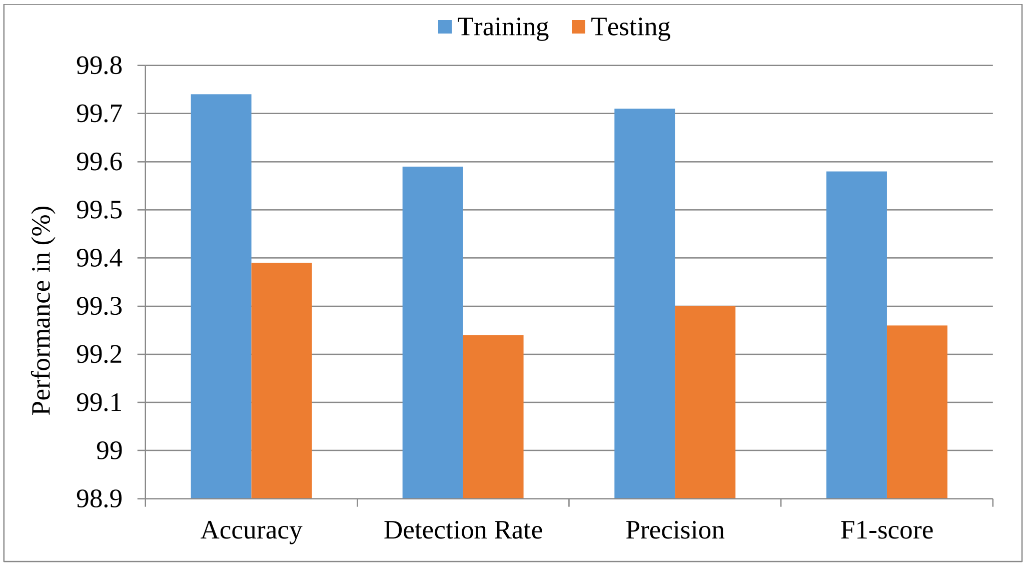

Fig. 5 Graphical illustration of BAOA-IESNN model’s binary classification results

Table 5 presents the binary classification results of the proposed BAOA-IESNN model using the BoT-IoT dataset. For this binary classification, the results were evaluated individually on both the training and test sets. The training set includes 80% of the data samples from the dataset, and the test set includes the remaining 20% of the data samples. The binary results were assessed using performance indicators like accuracy, DR, precision, and f1-score. Using the training set, the developed BAOA-IESNN model attained 99.74% accuracy, 99.59% DR, 99.71% precision, and 99.58% f1-score. Using the test set, the developed model attained 99.39% accuracy, 99.24% DR, 99.30% precision, and 99.26% f1-score. These binary results indicate that the model performed well and efficiently in both the training and testing. Compared to the training set's results, the test set's results are slightly lower. This difference in results was due to the fact that the model is optimized while training with the dataset. This causes minimized generalization to unknown test data. Figure 5 depicts the graphical chart of the BAOA-IESNN model's binary classification results.

Table 6. Multiclass classification results of the BAOA-IESNN model

Classes	Accuracy	DR	Precision	F1-score
Benign	99.25	99.08	99.18	99.12
DoS-HTTP	98.92	98.80	98.89	98.84
DoS-TCP	99.04	98.91	98.95	98.92
DoS-UDP	98.79	98.67	98.75	98.71
DDoS-HTTP	99.10	98.97	99.02	98.99
DDoS-UDP	98.82	98.68	98.79	98.74
DDoS-TCP	99.00	98.85	98.91	98.87
OS Fingerprinting	98.75	98.62	98.70	98.65
Service Scanning	98.89	98.73	98.86	98.80

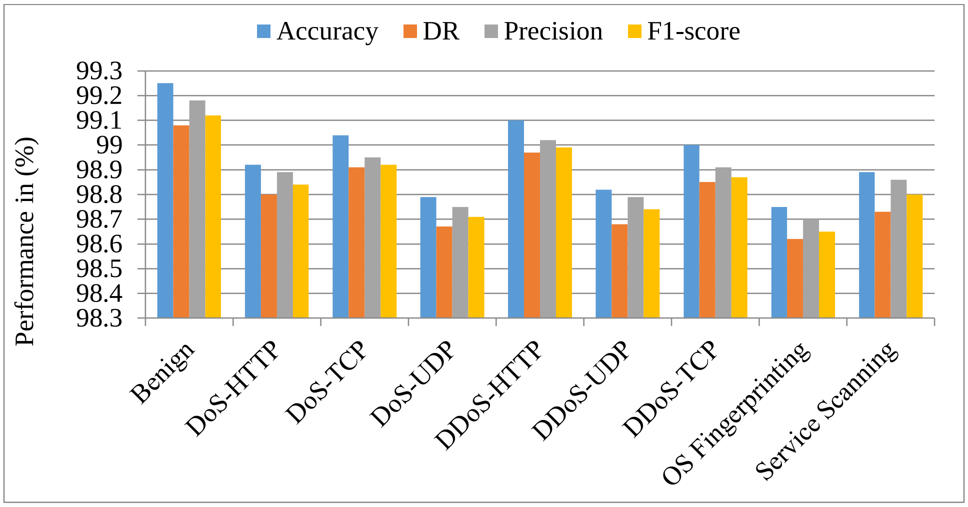

Fig. 6 Graphical illustration of BAOA-IESNN model's multiclass classification results

Table 6 presents the results of the BAOA-IESNN model's performance in multiclass classification. For this research, the multiclass classification performance was evaluated on 8 different attack classes and one benign class. The classified attack classes are DoS-HTTP, DoS-TCP, DoS-UDP, DDoS-HTTP, DDoS-UDP, DDoS-TCP, OS Fingerprinting, and Service Scanning.

The developed BAOA-IESNN model performed better in classifying the benign class with 99.25% accuracy, 99.08% DR, 99.18% precision, and 99.12% f1-score. Comparatively, the model performed well in detecting DoS-TCP, DDoS-HTTP, and DDoS-TCP attacks with 99.04%, 99.10%, and 99% accuracy.

Excluding the benign class, the developed model has attained an accuracy score ranging from 98.75% to 99.10% for all the attack classes. The model attained a detection rate ranging from 98.62% to 98.97% for all the attack classes. The model attained a precision score ranging from 98.70% to 98.95% for all the attack classes. The developed model achieved an F1-score ranging from 98.65% to 98.99% for all the attacks. The accuracy performance in multiclass classification indicates that the model can classify all the attacks accurately with a high degree of generalization. The detection rate of the developed model in detecting multiple attacks indicates that the model is effective in detecting positive occurrences of the attack classes.

The precision results of the model indicate that the model can minimize FPs while maintaining a higher prediction in positive classes. The F1-score of the model indicates that the model has a balanced performance in precision and recall. Overall, this multiclass classification performance highlights that the developed BAOA-IESNN model is better at detecting multiple attacks. Figure 6 depicts the graphical chart of the BAOA-IESNN model's performance in multiclass classification.

Table 7. Performance comparison with current models

Models	Accuracy	Detection Rate	Precision	F1-score
SAE [11]	96.00	94.00	98.00	96.00
AEFS-KENN [12]	98.12	96.30	95.85	96.97
CNN-LSTM [13]	98.00	NA	NA	98.00
LSTM [16]	91.20	92.50	NA	NA
EffiGRU-GhostNet [17]	97.80	97.89	97.92	97.90
RBSM [19]	97.81	97.80	97.92	97.82
SPWCC-DMN [20]	97.30	98.00	NA	NA
Spark-Chi-SVM [22]	96.80	94.36	94.36	NA
CNN-LSTM-PySpark [23]	98.75	98.75	98.75	98.75
RF+KNN [24]	98.23	98.89	98.81	98.95
ET+PCA [25]	98.37	90.38	96.81	90.38
CFS-SVM [26]	95.87	95.88	96.88	NA
Proposed Model	98.85	98.79	98.87	98.83

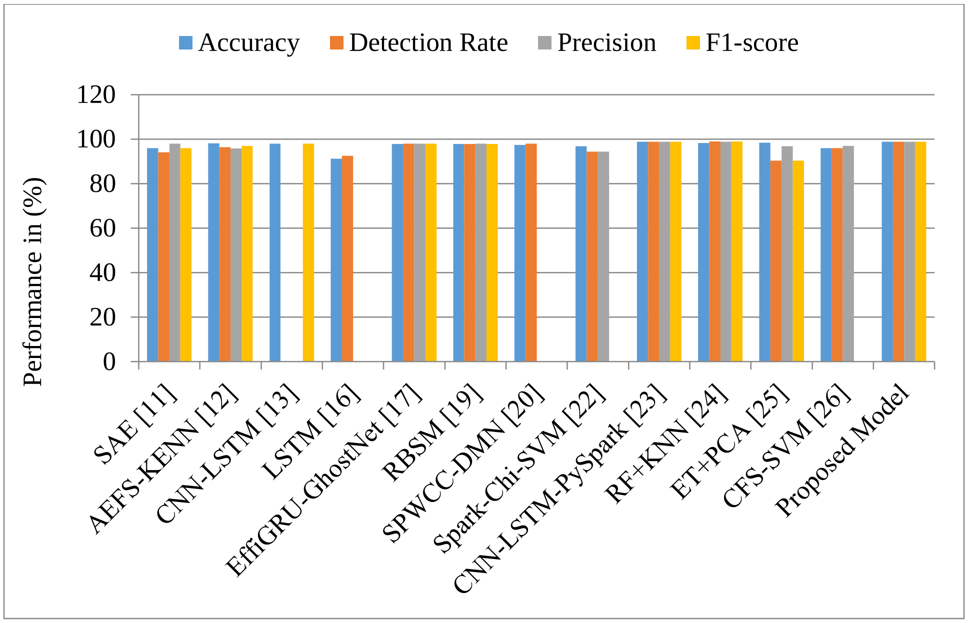

Fig. 7 Graphical illustration of results comparison

Table 7 presents a comparison of the results of the proposed BAOA-IESNN model's performance with the current models discussed in the related works section. For this comparison, the average score of the multiclass classification results is applied. This comparison shows that the results of the developed BAOA-IESNN model outperformed all the compared models in this research. The BAOA-IESNN model attained 98.85% average accuracy in multiclass classification, which is 0.1% to 7.65% higher than the other compared models in this research. Models like AEFS-KENN, CNN-LSTM, CNN-LSTM-PySpark, RF+KNN, and ET+PCA have attained a close accuracy performance compared to the proposed model within the 98% to 98.75% range. The LSTM [16] model is the least performed model with 91.20% accuracy. The BAOA-IESNN model attained 98.79% average detection rate in multiclass classification, which is 0.04% to 8.41% higher than the other compared models in this research, except the RF+KNN model. The RF+KNN model has achieved a 98.89% detection rate, which is 0.1% higher than the BAOA-IESNN model. The CNN-LSTM-PySpark has achieved a very close detection rate performance compared to the proposed model with 98.75 DR%. The models like EffiGRU-GhostNet, RBSM, and SPWCC-DMN have attained better performance within the range of 97.80% to 98%. The ET+PCA model is the least performed model with 90.38% DR.

The BAOA-IESNN model attained 98.87% average precision score in multiclass classification, which is 0.06% to 4.51% higher than the other compared models in this research. Models like SAE, CNN-LSTM-PySpark, and RF+KNN have attained very close performance within the range of 98% to 98.81%. The Spark-Chi-SVM model is the least performed model with 94.36% precision. The BAOA-IESNN model attained 98.83% average F1-score in multiclass classification, which is 0.08% to 8.45% higher than the other compared models in this research, except the RF+KNN model. The RF+KNN model has achieved 98.95% f1-score, which is 0.12% higher than the BAOA-IESNN model. The CNN-LSTM-PySpark has achieved a very close f1-score performance compared to the proposed model with 98.75 f1-score. The models like EffiGRU-GhostNet, RBSM, and CNN-LSTM have attained better performance within the range of 97.82% to 98%. The ET+PCA model is the least performed model with a 90.38% F1-score. Overall, based on this comparison, it is clear that the developed BAOA-IESNN model has outperformed all the compared models in the BD-based intrusion detection task.

Based on the obtained performance and results, the developed BAOA-IESNN model has several advantages. The advantages include the fact that the model is precise and efficient in handling IoT BD based on BAOA for feature selection and IESNN for attack classification. The Apache Spark applied in this research helped to enhance the model's processing capabilities and classification performance. The multiple processes in data preprocessing additionally enhance the classification performance in both binary and multiclass detection. However, the BAOA-IESNN model has a few limitations. The model has increased computational complexity and training time. This is due to the combination of the BAOA feature selection with SNN model.

5. Conclusion

This research proposed a BD-IoT intrusion detection model using the DL algorithm with Apache Spark for attack detection and classification. The developed intrusion detection model included multiple stages of processes such as data collection, data preprocessing, feature selection and classification. For this research, the BoT-IoT dataset was collected and applied for model training and evaluation. The collected dataset was processed in the preprocessing stage with multiple preprocessing methods like data cleaning, label encoding, RO, and min-max normalization. After preprocessing, the data set was divided into an 80:20 ratio. The majority of the data set was employed to train the model, and the remaining data was employed for evaluation. The BAOA technique was applied to choose the optimal features from the data set. Based on the selected optimal features, the IESNN model performed attack detection and classification. The IESNN model was effective in capturing temporal dependencies and dynamic patterns in the data. Finally, the performance of the BAOA-IESNN model was evaluated based on various performance indicators. The results were evaluated and compared with the current models for validation. The BAOA-IESNN model attained 99.39% accuracy, 99.24% detection rate, 99.30% precision, and 99.26% f1-score in binary classification, and 98.85% accuracy, 98.79% detection rate, 98.87% precision, and 98.83% f1-score in binary classification. These results highlight that the developed BAOA-IESNN model was highly accurate and effective in detecting intrusions on both binary and multiclass classifications. In future, the developed research model can be extended to apply and test alternative metaheuristic algorithms for effective FS. Additionally, the research can further focus on optimizing the computational complexity of the developed model. Furthermore, more diverse and recent IoT BD datasets can be applied and evaluated to improve the real-time adaptiveness and robustness.

Acknowledgments

The authors would like to thank the Sona College of Arts and Science, Salem, and Thiruvalluvar Government Arts College, Rasipuram, for their support in completing this project.

References

[1] Claudia Cavallaro et al., "Discovering Anomalies in Big Data: A Review Focused on the Applications of Metaheuristic and Machine Learning Techniques," *Frontiers in Big Data*, vol. 6, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[2] Joffrey L. Leevy, and Taghi M. Khoshgoftaar, "A Survey and Analysis of Intrusions Detections Model based on CSE-CIC-IDS 2018 Big Data," *Journal of Big Data*, vol. 7, no. 1, pp. 1-19, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[3] Luis Filipe Dias, and Miguel Coreia, "Big Data Analytic for Intrusions Detections: An Overview," *Handbooks of Research on Machine and Deep Learning Application for Cyber Security*, pp. 292-316, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[4] Xiao Chu et al., "Big Data and its V's with IoTs to Develop Sustainability," *Scientific Programming*, vol. 2021, no. 1, pp. 1-16, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[5] Noshina Tariq et al., "The Security of Big Data in Fog-Enabled IoTs Application Including Blockchains: A Survey," *Sensors*, vol. 19, no. 8, pp. 1-33, 2019. [CrossRef] [Google Scholar] [Publisher Link]

[6] KamtaNath Mishra et al., "Cloud and Big Data Security System Reviews Principle: A Decisive Investigation," *Wireless Personal Communication*, vol. 126, no. 2, pp. 1013-1050, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[7] Bijender Bansal et al., "Big Data Architectures for Networks Security," *Cyber Security and Networks Security*, pp. 233-267, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[8] Nnamdi Johnson Ogbuke et al., "Big Data Supply Chains Analytic: Ethical, Privacy and Security Challenges Posed to Business, Industries and Society," *Productions Planning & Controls*, vol. 33, no. 2-3, pp. 123-137, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[9] Prerna Agrawal, and Savita Gandhi, "Big Data Cyber Security Analytics," *Advanced Cyber Security Technique for Data, Blockchain, IoTs, and Network Protections*, pp. 21-48, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[10] S. Kumar Reddy Mallidi, and Rajeswara Rao Ramisetty, "Optimizing Intrusions Detections for IoTs: A Systematic Review of Machine Learning and Deep Learning Approach with Feature Selections and Data Balancing," *Wiley Interdisciplinary Review: Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 1-41, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[11] Isuru Udayangani Hewapathirana, "A Comparative Study of Two-Stage Intrusion Detection Using Modern Machine Learning Approach on the CSE-CIC-IDS 2018 Datasets," *Knowledge*, vol. 5, no. 1, pp. 1-19, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[12] Sankaramoorthy Muthubalaji et al., "An Intelligent Big Data Security Framework based on AEFS-KENN Algorithm for the Detection of Cyber-Attack from Smart Grids System," *Big Data Mining and Analytics*, vol. 7, no. 2, pp. 399-418, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[13] Abdulnaser A. Hagar, and Bharti W. Gawali, "Apache Sparks and Deep Learning Model for High-Performance Network Intrusion Detection using CSE-CIC-IDS 2018," *Computational Intelligences and Neurosciences*, vol. 2022, no. 1, pp. 1-11, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[14] Otmane Azeroual, and Anastasija Nikiforova, "Apache Spark and MLLIB-Based Intrusion Detection Systems or How the Big Data Technologies Can Secure The Data," *Information*, vol. 13, no. 2, pp. 1-18, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[15] Wei Zhong, Ning Yu, and Chunyu Ai, "Applying Big Data-Based Deep Learning Systems to Intrusion Detection," *Big Data Mining and Analytics*, vol. 3, no. 3, pp. 181-195, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[16] Mahzad Mahdavisharif, Shahram Jamali, and Reza Fotohi, "Big Data-Aware Intrusion Detection Systems in Communications Network: A Deep Learning Approach," *Journal of Grids Computing*, vol. 19, no. 4, pp. 1-28, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[17] Abdulrahman A. Alshdadi et al., "Big Data-Driven Deep Learning Ensemble for DDoS Attack Detection," *Future Internet*, vol. 16, no. 12, pp. 1-26, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[18] Md Abdur Rahman, and Hossain Shahriar, "Clustering Enable Robust Intrusions Detections Systems for Big Data Using Hadoop - PySpark," *2023 IEEE 20th International Conferences on Smart Communities: Improving Quality of Life using AI, Robotic and IoTs (HONET)*, Boca Raton, FL, USA, pp. 249-254, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[19] Rubayyi Alghamdi, and Martine Bellaiche, "Evaluations and Selections Model for Ensembled Intrusions Detections System in IoTs," *IoT*, vol. 3, no. 2, pp. 285-314, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[20] Brunel Elvire Bouya-Moko, Edward Kwadwo Boahen, and Changda Wang, "Fuzzy Local Information and Bhattacharya-Based C-Mean Clustering and Optimized Deep Learning in Spark Frameworks for Intrusion Detection," *Electronics*, vol. 11, no. 11, pp. 1-16, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[21] Abhijit Dnyaneshwar Jadhav, and Vidyullatha Pellakuri, "Highly Accurate and Efficient Two Phase-Intrusion Detection System (TP-IDS) using Distributed Processing of HADOOP and Machine Learning Techniques," *Journal of Big Data*, vol. 8, no. 1, pp. 1-22, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[22] Suad Mohammed Othman et al., "Intrusions Detections Models using Machine Learning Algorithms on Big Data Environments," *Journal of Big Data*, vol. 5, no. 1, pp. 1-12, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[23] Sami Yaras, and Murat Dener, "IoTs-Based Intrusion Detection Systems using New Hybrid Deep Learning Algorithms," *Electronics*, vol. 13, no. 6, pp. 1-28, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[24] Siriporn Chimphlee, and Witcha Chimphlee, "Machine Learning to Improve the Performances of Anomaly-Based Networks Intrusions Detections in Big Data," *Indonesian Journal of Electrical Engineering and Computer Sciences*, vol. 30, no. 2, pp. 1106-1119, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[25] Md. Alamin Talukder et al., "Machine Learning-Based Networks Intrusions Detections for Big and Imbalanced Data using Oversampling, Stacking Features Embedding and Feature Extraction," *Journal of Big Data*, vol. 11, no. 1, pp. 1-44, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[26] Anita Shiravani, Mohammad Hadi Sadreddini, and Hassan Nosrati Nahook, "Networks Intrusions Detections using Data Dimension Reductions Technique," *Journal of Big Data*, vol. 10, no. 1, pp. 1-25, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[27] Xiangyu Liu, and Yanhui Du, "Toward Effective Features Selections for IoTs Botnet Attacks Detections using a Genetic Algorithm," *Electronics*, vol. 12, no. 5, pp. 1-12, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[28] Abdelaziz Al Dawi et al., "An Approach to Botnet Attack in the Fog Computing Layers and Apache Spark for Smart Cities," *The Journal of Supercomputing*, vol. 81, no. 4, pp. 1-30, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[29] Yujie Zhang, and Zebin Wang, "Feature Engineering and Model Optimizations-Based Classifications Methods for Network Intrusion Detection," *Applied Sciences*, vol. 13, no. 16, pp. 1-25, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[30] Qinglei Yao, and Xiaoqiang Zhao, "An Intrusions Detections Imbalanced Data Classifications Algorithm based on CWGAN-GP Oversampling," *Peer-to-Peer Networking and Applications*, vol. 18, no. 3, pp. 1-16, 2025. [CrossRef] [Google Scholar] [Publisher Link]

[31] Aya G. Ayad, Nehal A. Sakr, and Noha A. Hikal, "A Hybrid Approach for Efficient Features Selections in Anomaly Intrusion Detection for IoTs Networks," *The Journal of Supercomputing*, vol. 80, no. 19, pp. 26942-26984, 2024. [CrossRef] [Google Scholar] [Publisher Link]

[32] Imène Neggaz, and Hadria Fizazi, "An Intelligent Handcrafted Features Selections using Archimedes Optimizations Algorithms for Facial Analysis," *Soft Computing*, vol. 26, no. 19, pp. 10435-10464, 2022. [CrossRef] [Google Scholar] [Publisher Link]

[33] Lingling Fang, Yutong Yao, and Xiyue Liang, "New Binary Archimedes Optimizations Algorithms and its Applications," *Expert Systems with Applications*, vol. 230, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[34] B. Meenakshi, and D. Karunkulali, "Enhanced Elman Spikes Neural Networks for Clusters Heads-Based Energy Aware Routing in WSNs," *Transaction on Emerging Telecommunication Technologies*, vol. 34, no. 3, 2023. [CrossRef] [Google Scholar] [Publisher Link]

[35] M. Deepanayaki, and Vidyaatulasiraman, "Enhanced Elman Spikes Neural Networks Optimized with Red Fox Optimizations Algorithms for Sugarcanes Yield Grades Predictions," *Smart Science*, vol. 11, no. 3, pp. 568-582, 2023. [CrossRef] [Google Scholar] [Publisher Link]