Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.043752

ARTICLE

Intrusion Detection System with Customized Machine Learning Techniques for NSL-KDD Dataset

Mohammed Zakariah[1], Salman A. AlQahtani[2,*], Abdulaziz M. Alawwad[1] and Abdullilah A. Alotaibi[3]

[1]Department of Computer Science, College of Computer and Information Science, King Saud University, Riyadh, 11495, Saudi Arabia

[2]New Emerging Technologies and 5G Network and Beyond Research Chair, Department of Computer Engineering, College of Computer and Information Sciences, King Saud University, Riyadh, 11495, Saudi Arabia

[3]Department of Computer Engineering, College of Computer and Information Science, King Saud University, Riyadh, 11495, Saudi Arabia

*Corresponding Author: Salman A. AlQahtani. Email: salmanq@ksu.edu.sa

Received: 11 July 2023 Accepted: 18 October 2023 Published: 26 December 2023

ABSTRACT

Modern networks are at risk from a variety of threats as a result of the enormous growth in internet-based traffic. By consuming time and resources, intrusive traffic hampers the efficient operation of network infrastructure. An effective strategy for preventing, detecting, and mitigating intrusion incidents will increase productivity. A crucial element of secure network traffic is Intrusion Detection System (IDS). An IDS system may be host-based or network-based to monitor intrusive network activity. Finding unusual internet traffic has become a severe security risk for intelligent devices. These systems are negatively impacted by several attacks, which are slowing computation. In addition, networked communication anomalies and breaches must be detected using Machine Learning (ML). This paper uses the NSL-KDD data set to propose a novel IDS based on Artificial Neural Networks (ANNs). As a result, the ML model generalizes sufficiently to perform well on untried data. The NSL-KDD dataset shall be utilized for both training and testing. In this paper, we present a custom ANN model architecture using the Keras open-source software package. The specific arrangement of nodes and layers, along with the activation functions, enhances the model's ability to capture intricate patterns in network data. The performance of the ANN is carefully tested and evaluated, resulting in the identification of a maximum detection accuracy of 97.5%. We thoroughly compared our suggested model to industry-recognized benchmark methods, such as decision classifier combinations and ML classifiers like k-Nearest Neighbors (KNN), Deep Learning (DL), Support Vector Machine (SVM), Long Short-Term Memory (LSTM), Deep Neural Network (DNN), and ANN. It is encouraging to see that our model consistently outperformed each of these tried-and-true techniques in all evaluations. This result underlines the effectiveness of the suggested methodology by demonstrating the ANN's capacity to accurately assess the effectiveness of the developed strategy in identifying and categorizing instances of network intrusion.

KEYWORDS

Artificial neural networks; intrusion detection system; classification; NSL-KDD dataset; machine and deep-learning; neural network

1 Introduction

Network communication is increasingly the target of threats and attacks as it is used more often across all sectors of the economy. Sorting network data into regular or suspect categories is a crucial step in trying to stop such assaults. "Irregularity detection" refers to this task, which deals with unlikely events in network communication. The incident will be correctly labeled as anomalous and treated with suspicion if there is a significant deviation from the standard [1]. The score will reflect how the new occurrence differs from the norm. In Machine Learning (ML), we often search for a broad solution space for the best model that fits the data. The term "solution space" in the context of Artificial Neural Network (ANN) refers to the space of all approximated or precise functions that an ANN can represent.

Modern networks are now vulnerable to a variety of threats due to the constant increase in internet-based traffic, making them susceptible to various malicious activities. In addition to interfering with the efficient operation of network infrastructure, intrusive network traffic also wastes time and money. To prevent, detect, and mitigate intrusion incidents in this dynamic environment, the development of strong and efficient Intrusion Detection System (IDS) strategies has become essential. The nature of network threats is constantly evolving, making it necessary to adopt cutting-edge ML techniques to strengthen the security ecosystem. Traditional approaches frequently fall short in this regard [2]. The complexity of intrusion incidents has increased as cyber attackers use more advanced techniques, necessitating solutions that go beyond conventional rule-based systems. IDS, a pillar of network security, is essential for quickly spotting and responding to suspicious or malicious network activity. To create IDS systems that are more adaptable and resilient, cutting-edge ML techniques, like ANNs, must be incorporated. ANNs have demonstrated their ability to identify complex patterns and anomalies within enormous and complex network datasets, making them especially well-suited for addressing the dynamic issues brought on by changing network threats.

It is crucial to adapt these methods to the unique requirements of intrusion detection to fully realize the potential of machine learning. It is possible to improve the precision and effectiveness of threat detection by creating specialized ML techniques that are tailored for intrusion detection scenarios. With this strategy, the IDS can adjust to changing attack methodologies, guaranteeing the prompt detection of new threats and reducing false positives [3]. The combination of cutting-edge ML methods and IDS is in line with the requirement for proactive security measures that keep up with malicious actors. Organizations can better protect their networks from a wide range of intrusion attempts by implementing ML-driven IDS strategies. To reduce the risks brought on by evolving network threats and guarantee the ongoing integrity of network infrastructure, it is crucial to pursue innovative and adaptable IDS solutions.

Artificial Intelligence (AI), often known as ML, is the study of methods that give computers the ability to learn. ML techniques, like ANN, typically learn from data samples to classify or detect patterns in the data, and they then make it possible for computer systems to anticipate new or unforeseen data instances based on the discovered patterns [2]. Depending on the learning process, ML shall be classified into two main categories: supervised learning and unsupervised learning. Supervised learning identifies the patterns needed to map an input to an output from pairs of data samples with labeled input outputs [3]. The classification problem is a supervised learning problem successfully applied to tackle Network-based Intrusion Detection System (NIDS) issues like those in [4]. Finding a mapping that can identify a hidden structure from unlabelled data samples is the goal of unsupervised learning. The unlabelled data samples are an effective tool for detecting structures [5]. Due to the flexibility of the requirement for labels on training data in unsupervised learning, many unsupervised learning techniques, including clustering-based NIDS [5] and self-organizing map-based NIDS, have also been widely used for NIDS difficulties. When an IDS simultaneously achieves high detection accuracy and low false-positive rates, it is said to be effective or precise in spotting interruptions [6]. Because of IDS, the scientific community has witnessed a new revolution. A Deep Learning (DL) approach may take atypical network packet behavior into account for labeled and unlabelled data. The most widely used ML approaches for identifying malicious discoveries are k-Nearest Neighbors (KNN), Random Forest (RF), Support Vector Machine (SVM), fuzzy interpolation, ANN, Multilayer Perceptron (MLP), and others [7,8]. ANN is a term used to describe a group of algorithms that teach AI to make new things. It resembles how the human brain is structured and how the mind organizes learning. To detect unauthorized activities in network traffic, ANN is actively submitting applications. The authors of [7] presented ANN-based threat discovery using the UNSW15 dataset to address verification issues in IoT contexts and achieved 84 percent detection accuracy. Nwakanma et al. [9] represented two ANN models using the NSL-KDD dataset and discovered identification accuracy of 93.98% and 84.05%, respectively.

One of the most popular optimizers, Stochastic Gradient Descent (SGD) is frequently used to reduce the loss function. One can look for a solution by moving counterclockwise around the gradient of the loss function while using SGD as an optimizer. This learning technique may overfit the learning model and significantly affect the generalization error or performance on unknown data due to the complexity and richness of the solution space, even while it still yields positive results on training data [10]. In ML, regularisation solves this issue and stops the learning model from getting too complicated.

In Neural Network (NN), a dropout is the most common regularization technique, but early halting is the most common regularization technique in iterative learning [11]. In statistics and machine learning, the term "regularisation" is frequently used about the "loss" or "error" function. This method has the advantage of including model complexity in the function, which needs to be optimized. SVMs [12,13] are algorithm that uses various principles to solve optimization problems. The flowchart in Fig. 1 depicts the suggested IDS model utilizing the ANN method and the NSL-KDD dataset.

Data pre-treatment is followed by data import. The NSL-KDD data collection consists of one class attribute and 41 other attributes. Of those 41 qualities, some play no part at all, and others hardly play a part in detecting attacks. The operations of feature reduction and normalization start if the answer is yes. The ANN architecture is first picked. After that, a neural network is trained with the appropriate training data. The testing dataset and testing neural network are then shown [14]. Several variables shall be used to assess NN's performance. Typical metrics include classification, detection rate, accuracy, and false-positive rate.

The present regularisation algorithms either perform poorly due to a deficit of relevant data or do not encourage sparsity in challenging scenarios when the number of attributes is more than the number of observations and they are correlated [14]. To find the optimal solution within an ample solution space, this endeavor will put into practice an original regularisation technique that considers the link among the weight matrix entries. After that, the space restriction was enhanced and might be controlled by restricting and enlarging this area in line with the severity of the penalty. As a result, it makes it possible to identify the learning model that is a minor complex. Furthermore, we intend to analyze the algorithm from a multiclass classification perspective [15–17] to distinguish between legitimate and varied harmful connections. Using the standard deviation to decay the weight matrices to derive the regularisation term, the IDS technique described in this research also includes innovative regularisation design considerations.

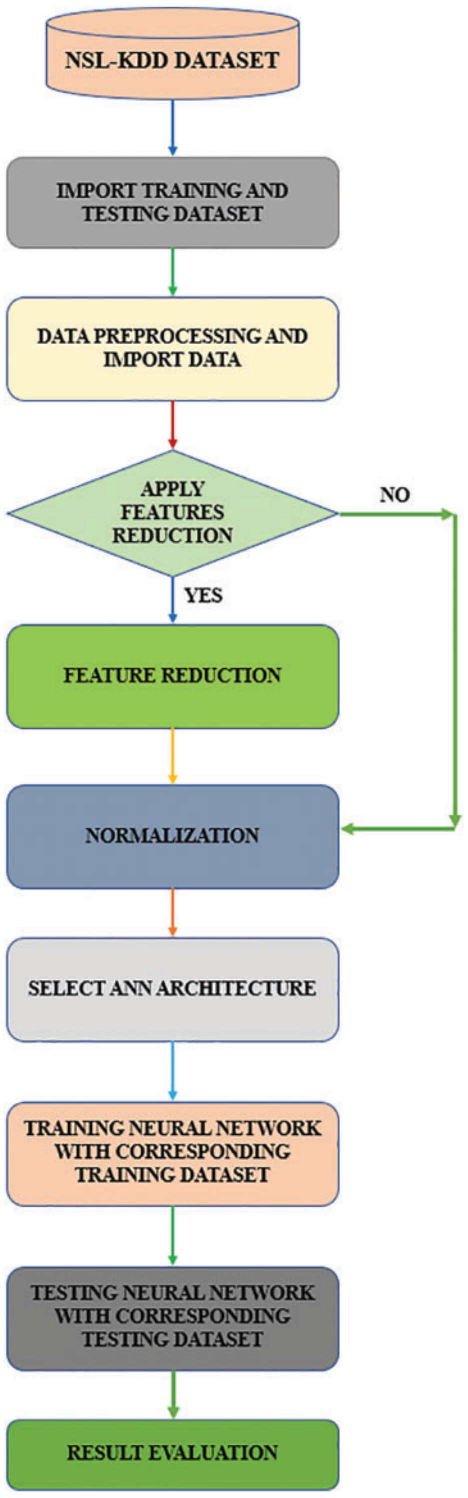
NSL-KDD DATASET
IMPORT TRAINING AND TESTING DATASET
DATA PREPROCESSING AND IMPORT DATA
APPLY FEATURES REDUCTION
NO
YES
FEATURE REDUCTION
NORMALIZATION
SELECT ANN ARCHITECTURE
TRAINING NEURAL NETWORK WITH CORRESPONDING TRAINING DATASET
TESTING NEURAL NETWORK WITH CORRESPONDING TESTING DATASET
RESULT EVALUATION

Figure 1: Flowchart of the proposed IDS model using ANN architecture

We combined the recommended regularizer with an ANN model for classification tasks and evaluated its performance in identifying anomalies in contrast to well-known regularisation techniques using the NSL-KDD dataset with unique training and testing sets [18,19]. The following is a summary of this paper's significant contributions:

- This study proposes a novel IDS based on ANNs using the NSL-KDD data set. Because of this, the ML model generalizes well enough to perform admirably on untested data.
- We present a custom ANN model architecture using the Keras open-source software package. The specific arrangement of nodes and layers, along with the activation functions, enhances the model's ability to capture intricate patterns in network data.
- The NSL-KDD dataset is introduced in this paper as a replacement for the KDD Cup'99 dataset, addressing its flaws and enhancing data quality.
- To significantly reduce dimensionality and preserve as much variability as possible, the PCA technique is used. The selection of core features according to the variance they explain improves data representation, allowing IDS to process data quickly.
- We compared our proposed model to widely used benchmark techniques, including decision classifier combinations and ML classifiers. It is encouraging to see that in every evaluation, our model consistently outperformed each of these tried-and-true methods.

The structure of this study is as follows: Section 2 discusses a list of recent research on the IDS for the NSL-KDD dataset using the ANN model, while Sections 3 and 4 describe our datasets and provide methodology. After that, Section 5 presents the analysis and findings. Finally, Section 6 discusses the results, and Section 7 presents the conclusion concerning our research.

2 Literature Review

Various writers discovered a wide range of alternative ML methods for identifying network anomalies and threats. According to earlier research, another crucial role of a network detection system is the detection of ongoing intrusions into secured networks. This section explains the viewpoints and concepts discussed in earlier research about NIDS methods using ML techniques and the NSL-KDD dataset. Deep packet inspection-based intrusion detection systems can employ Marwan and Binsawad's [1] innovative method of leveraging ANN to identify malicious network traffic. Using ANN for wired LANs, the results demonstrate that this innovative classification approach can identify DoS attacks. The suggested ANN classifier performs with 96% accuracy on the training data set. IDS model was designed and trained to utilize a variety of DNN architectures [3], including CNN, autoencoders, and RNN. After training on the NSLKDD training dataset, these deep models were evaluated using the NSLKDD test datasets, NSLKDDTest21 and NSLKDDTest+. To improve the model's validity, we combined classic ML models with several well-known classification techniques, including RF, SVM, NB, KNN, and Decision Tree. The RoC curve, the precision-recall curve, and the area under the RoC curve, which represent average accuracy and precision of classification, were employed to evaluate both DNN and conventional ML models. The accuracy for the test dataset was 86% for the DCNN model and 88% for the LSTM model, respectively.

This proved that DL is a technology that is both useful and prospective for use in information security applications as well as other application domains.

The CIDDS-001 dataset gets evaluated stochastically and computationally in [4]. The authors of CIDDS-001 have developed an IDS utilizing the KNN classifier. Their system achieved a minimum accuracy of 99.2% with 2NN and a maximum accuracy of 99.7% with 5NN. When categorizing traffic using the same dataset, the authors of [6] conducted an analytical study to assess the efficacy of KNN and k-Means clustering algorithms. Both techniques have an accuracy of over 98%. Three separate datasets—GPRS, NSL-KDD, and UNSW-NB15—were employed with either 10-fold cross-validation or randomization in the anomaly-based IDS recommended by the authors in [7]. The authors of [8] suggested an improved IDS based on hybrid feature selection and two-level classifier ensembles. Based on statistics and significance tests on the NSL-KDD dataset, the proposed classifier shows 85.8% accuracy, 86.8% sensitivity, and an 88.0% detection rate.

Fig. 2 shows the NSL-KDD dataset's structure and numerous feature selection methods. The researchers then employ FS approaches, such as J48, RF, ANN, CNN, DNN, SVM, DT, and KNN, depending on their dataset. These techniques were retrieved using metrics after a 10-fold cross-validation procedure. Moreover, using the KDD Cup'99 dataset for their test, the authors found that the CANN classifier outperformed SVM and KNN by a similar or marginally more significant margin. Using fewer computer resources, they effectively detected threats using a hybrid principal component analysis technique, and analyzed the NSL-KDD dataset using various ML techniques for IDS.

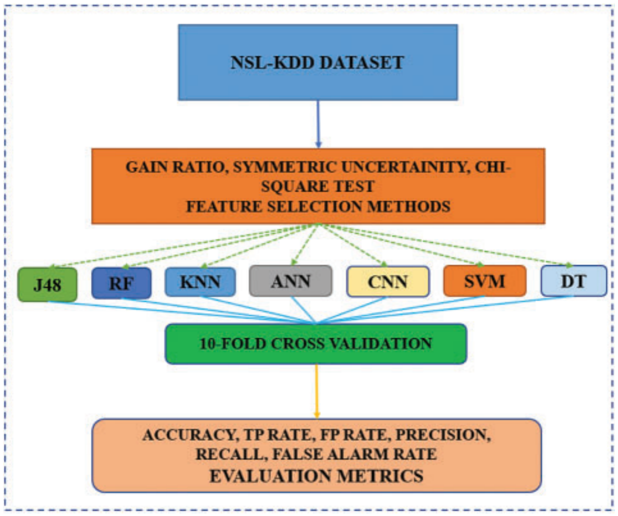
NSL-KDD DATASET
GAIN RATIO, SYMMETRIC UNCERTAINITY, CHI-SQUARE TEST
FEATURE SELECTION METHODS
J48
RF
KNN
ANN
CNN
SVM
DT
10-FOLD CROSS VALIDATION
ACCURACY, TP RATE, FP RATE, PRECISION, RECALL, FALSE ALARM RATE
EVALUATION METRICS

Figure 2: Framework of NSL-KDD dataset with different feature selection methods

The authors in [9] developed a novel model for IDS by combining the various categorization capabilities of NN and fuzzy logic. Adaptive IDS using naive Bayesian and boosted classifiers was proposed as a new learning method [10]. They also ran an experiment using data from the KDD Cup'99. According to the experiment results, the suggested strategy had much lower false-positive rates and higher detection rates for different forms of network penetration. For feature selection and weighting, the author in [11] recommended combining GA and KNN. The recommended model detected DoS attacks on the KDD Cup'99 dataset. The results showed that while accuracy was determined to be 97.24% for known assaults, it was discovered to be 78% for unknown attacks. Based on the Michigan, IRL, and Pittsburgh approach, the authors of [12] presented three different types of fuzzy genetic systems for IDS. In [13], a fresh feature representation approach was presented. The CANN approach measured and added the distances between each piece of data and its closest neighbor, as well as the distance between a sample of data and the cluster's nucleus.

A DNN, a sort of DL model, was investigated in the study [20] to create a flexible and effective IDS to identify and categorize unanticipated and unpredictable intrusions. The fast growth of attacks and the ongoing change in network behavior need the evaluation of multiple datasets that have been produced over time using both static and dynamic methods. This kind of research makes it easier to choose the optimal algorithm for reliably identifying upcoming cyberattacks. Chkirbene et al. looked into the NSL-KDD dataset for IDS that is classification-based [21]. This study investigated the ability of several classification algorithms to recognize odd network traffic patterns using the NSL-KDD dataset. They examined how the network protocol and the attacks resulting in unusual network traffic are related.

Ismail et al. [22] developed a hybrid attack detection technique in the same context to recognize network attacks by fusing decision trees, RF, and SVM. These techniques have lower detection rates and higher false alarm rates since they cannot prevent every intrusion attempt. An effective ML ensemble strategy for IDS that prioritizes increasing detection quality was put forth by the author in the study [23]. To improve the detection of various intrusion classes, including unique assaults like R2L and U2R, they stressed the important tweaking of ML model parameters and pre-processing techniques. The utilization of two widely-used datasets, KDD Cup'99 and NSL-KDD, with data augmentation to rebalance them, is one of the suggested methodology's many important benefits. They created a specific classifier architecture and established a three-step process using MLP in a cascaded structure. This method greatly enhanced detection quality and increased accuracy.

Adeel et al. [24] additionally used the unlabelled data by fusing the NB classifier with the tried-and-true EM method. Other researchers have employed the NB and decision tree algorithms for NIDS, which provided good accuracy for various networks [24]. None of the initiatives, however, emphasize the usage of merged classes; instead, they all rely on the results of separate classifiers. As a result, attack detection might be less dependent on training data, easier to develop, and quicker to evaluate if there is a gap between classes. As a result, performance might be enhanced by using FS and classifier techniques [25]. The use of classifiers in conjunction with the three FS is a recurring theme in our work. The lengthier training times needed for big data sets limit the utility of SVM and ANN, even though they outperformed other classification models in our study. Multiple FS techniques are integrated with ANN and SVM classifiers to detect attacks accurately.

The goal of the study [26] was to develop a method for predicting intrusions that may foretell botnet attacks on AGVs. The N-Balo dataset is utilized in this work for classification, clustering, and prediction. The method used in this study might serve as a foundation for creating the most dependable and highly secure AGV network. A Crow-Search-based ensemble classifier is utilized in the study [27] to categorize an IoT-based UNSW-NB15 dataset. First, the dataset's most important characteristics are identified using the Crow-Search method, and then these features are supplied to an ensemble classifier for training that uses the Linear Regression, Random Forest, and XGBoost algorithms. The study [28] provided a thorough description of the use of FL in various anomaly detection contexts. The associated FL implementation issues are also noted, which gives insight into the potential areas for future study. According to the study [29], a novel feature extraction and classification method for safe data transfer and intrusion detection in a biometric authentication system was presented. Here, an intrusion is discovered by gathering the IoT-based smart building's biometric information. The processing of this biometric data includes noise reduction, smoothing, and normalization.

Table 1 contains a list of references to earlier studies, together with information about their methodology, outcomes, and conclusions.

Table 1: Comparative analysis of intrusion detection methods and results

Reference	Experimental parameters/dataset	Methodology	Results	Observations/remarks
[1]	Dataset NSL-KDD. Dataset CIDDS-001, UNSW-NB15	Machine Learning (ML), Principle Component Analysis (PCA), Recurrent Neural Network (RNN), Particle Swarm Optimization (PSO), Support Vector Machine (SVM), k-Nearest Neighbour (KNN), and multiclass classification	On the UNSW-NB15, NSL-KDD, and CIDDS-001 datasets, accuracy using 10-fold cross-validation was 98.53%, 94.58%, and 97.87%, respectively.	Demonstrates strong accuracy across diverse datasets; considerations for computational complexity may arise.
[2]	Dataset CIDDS-001, NSLKDD	Convolutional Neural Network (CNN), Random Forest (RF), Deep Neural Network (DNN), Deep Learning (DL), RNN, and Machine Learning (ML). ML classifier stacking	Detect attacks with an accuracy of up to 97.99%.	Highlights the synergy of multiple classifiers for improved detection accuracy; potential implications for model interpretability.
[5]	UNSW-NB15 and NSL-KDD datasets	Area Under the Curve (AUC), Gradient Boosting Machine (GBM), fuzzy, two-tier, and tree-based classifier ensemble	99.1% accuracy in attack detection.	Illustrates the strength of ensemble techniques in attaining robust and high detection accuracy.
[10]	We have taken 38 attributes out of the network meta-data.	Clustering, feature analysis, parameter selection, and anomaly detection are all part of fuzzy C-means clustering	High accuracy of 98% was attained.	Successful utilization of clustering-based techniques for accurate intrusion detection and feature extraction.
[19]	UNSWNB15, CICIDS2017, NSL-KDD datasets and KDD Cup'99	Wireless Sensor Network (WSN), Extreme Gradient Boosting (XGBoost), RF, K-NN, SVM, ANN, RNN, DNN, ML, DL, Signature based Intrusion Detection System (SIDS), Anamoly Intrusion Detection System (AIDS)	About 80% of the most useful solutions were based on DNN and ANN algorithms.	Highlights the growing significance of DNN and ANN algorithms in intrusion detection; considerations for algorithm selection may be relevant.
[30]	Datasets UNSW-NB15 and NSL-KDD	XGBoost, Long Short-Term Memory (LSTM), ML, RNN, and Gated Recurrent Unit (GRU).	85.14 percent TAC accuracy for XGBooost-LSTM. 99.50% Validation Accuracy.	Points to the value of hybrid models like XGBoost-LSTM in bridging traditional ML and deep learning approaches.

(Continued)

Table 1 (continued)

Reference	Experimental parameters/dataset	Methodology	Results	Observations/remarks
[31]	KDD Cup'99 dataset for NSL-KDD	Decision Tree, NB, KNN, ML, DL, FFDNNs, SVM	Compared to previous approaches, FFDNN-IDS delivers an increase in accuracy of roughly 86.53%.	Emphasizes the role of feature selection techniques like FFDNNs in enhancing accuracy; considerations for computational complexity may arise.

The literature review highlights the contributions this study makes to the field by highlighting several research gaps and limitations that are addressed in this study. Previous research, particularly when using datasets like NSL-KDD, lacks a thorough comparison of different machine learning techniques for intrusion detection. This study fills in the gap by presenting a variety of ML techniques and evaluating their performance on the same dataset, including CNN, RF, DNN, DL, RNN, SVM, and KNN. This method enables a more insightful evaluation of their effectiveness in detecting network anomalies. Second, it is clear that ensemble techniques have not been thoroughly explored. This study introduces the idea of classifier stacking, which combines different Machine Learning (ML) models to improve attack detection precision. This addresses the difficulty of using just one model and emphasizes the benefit of fusing various points of view. The study sets itself apart by advocating hybrid attack detection techniques that maximize detection while minimizing false alarms, such as decision trees, RF, and SVM. This novel method may close the gap between the shortcomings of various models, increasing accuracy. This research emphasizes the significance of taking into account merged classes in light of the shortcomings in previous approaches' failure to take into account merged attack patterns. By doing this, intrusion detection systems' ability to respond to new threats is improved. This study integrates multiple feature selection techniques with these classifiers to address big data challenges, especially in algorithms like SVM and ANN. Their performance and suitability for large datasets are intended to be improved by this strategy. Furthermore, this research uses a variety of metrics, such as accuracy, AUC curve, and precision-recall curve, to provide a more thorough evaluation of suggested approaches, whereas other studies may lack thorough evaluation metrics. Direct comparisons are hampered by the variation in dataset usage across previous studies. By consistently using the NSL-KDD dataset for both training and testing, this research reduces this by enabling more accurate and insightful comparisons between various approaches.

3 Dataset

The 2009 NSL-KDD datasets were used in research on intrusion detection. The dataset is made up of the KDDTrain dataset, which serves as the testing set, and the KDDTest+ and KDDTest21 datasets, as shown in Table 2.

The following are some features of the NSL-KDD collection over the KDD Cup'99 sample:

1. Because of the large number of records in the training and testing datasets, experiments will be conducted.
2. The number of titles selected in each group of difficulty levels is inversely related to the number of records in the base KDD data set.

3. As a result, the method performs better than ML strategies over a more extensive range, improving the accuracy of evaluating other learning techniques adequately. Furthermore, since there are enough observations in the training and evaluation sets, it is possible to run the tests on the complete collection rather than selecting a small sample randomly.

Table 2: NSL-KDD dataset collection for intrusion detection training and testing

	Total	Normal	DOS	Probe	R2L	U2R
KDD Train+	125973	67343	45827	11456	995	49
KDD Test+	25192	13449	9234	2289	209	11
KDD Test-21	22542	12709	7749	1867	175	42

As a result, the evaluation outcomes of numerous study endeavors will be equivalent and consistent. As shown in Fig. 3, the attack classes in the NSL-KDD collected data are divided into four categories.

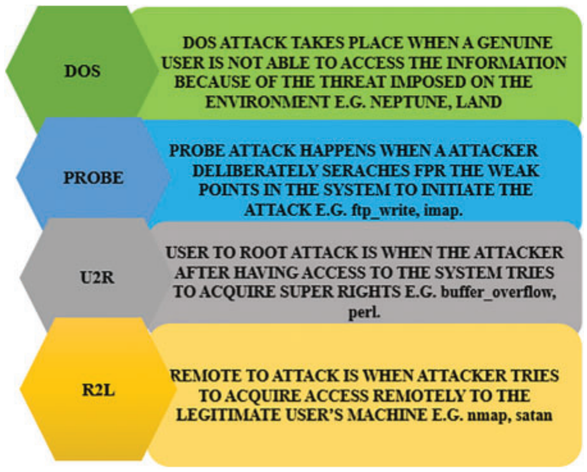
DOS
DOS ATTACK TAKES PLACE WHEN A GENUINE USER IS NOT ABLE TO ACCESS THE INFORMATION BECAUSE OF THE THREAT IMPOSED ON THE ENVIRONMENT E.G. NEPTUNE, LAND
PROBE
PROBE ATTACK HAPPENS WHEN A ATTACKER DELIBERATELY SERACHES FPR THE WEAK POINTS IN THE SYSTEM TO INITIATE THE ATTACK E.G. ftp_write, imap.
U2R
USER TO ROOT ATTACK IS WHEN THE ATTACKER AFTER HAVING ACCESS TO THE SYSTEM TRIES TO ACQUIRE SUPER RIGHTS E.G. buffer_overflow, perl.
R2L
REMOTE TO ATTACK IS WHEN ATTACKER TRIES TO ACQUIRE ACCESS REMOTELY TO THE LEGITIMATE USER'S MACHINE E.G. nmap, satan

Figure 3: Attack classes in the NSL-KDD dataset

Every data characteristic must transform being added to the algorithm. In this inquiry, data pre-processing is an essential stage. Outliers should be eliminated during data preparation to reduce the data collection size. The procedure entails replacing outlier data or reducing outlier influence through adjustments to outlier weights. Fig. 4's train data and test data representation below demonstrate how this uses reliable algorithms to determine the importance of outliers.

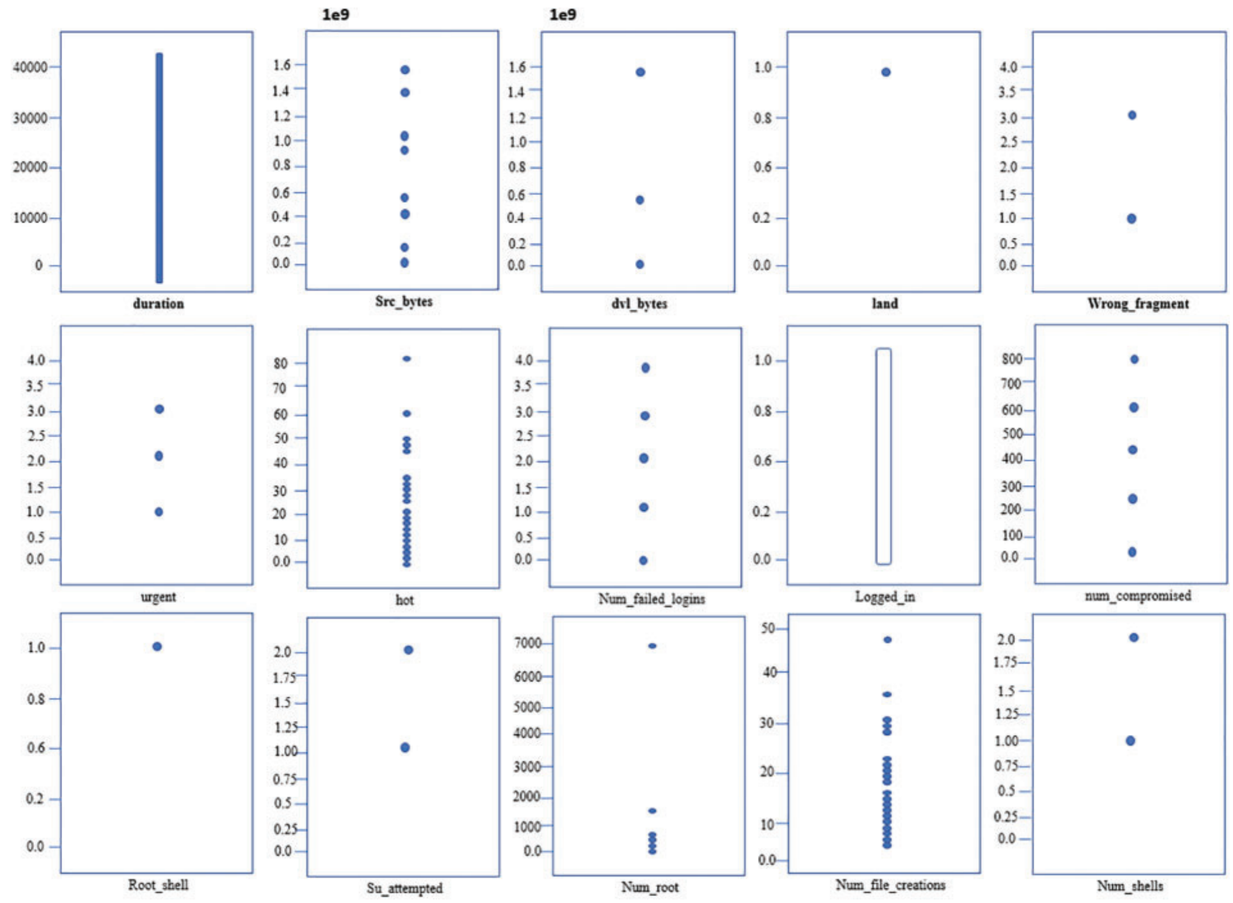

Figure 4: Train data outlier

Fig. 5 shows the outlier in the test data. Depending on the distribution of the underlying data, an observation may be labeled as an outlier. This section's content is limited to univariate data sets assumed to have a roughly normal distribution. If the normality assumption for the data under consideration is incorrect, the appearance of an outlier may be more due to the data's non-normality than to the assumption that it is normal.

In the subsequent phase, we used categorical characteristics, which can only have a limited range of values. Select a protocol type with its occurrences in Fig. 6 to explore the assortment of protocol types present in the NSL-KDD dataset.

The services were then examined to determine how many were utilized in our dataset, as depicted in Fig. 7. Despite being more challenging to manage than numerical data, categorical data shall be included in ML tasks. To use any of these strategies, categorical qualities must first be translated into numerical features because many ML algorithms, in particular, require numerical input.

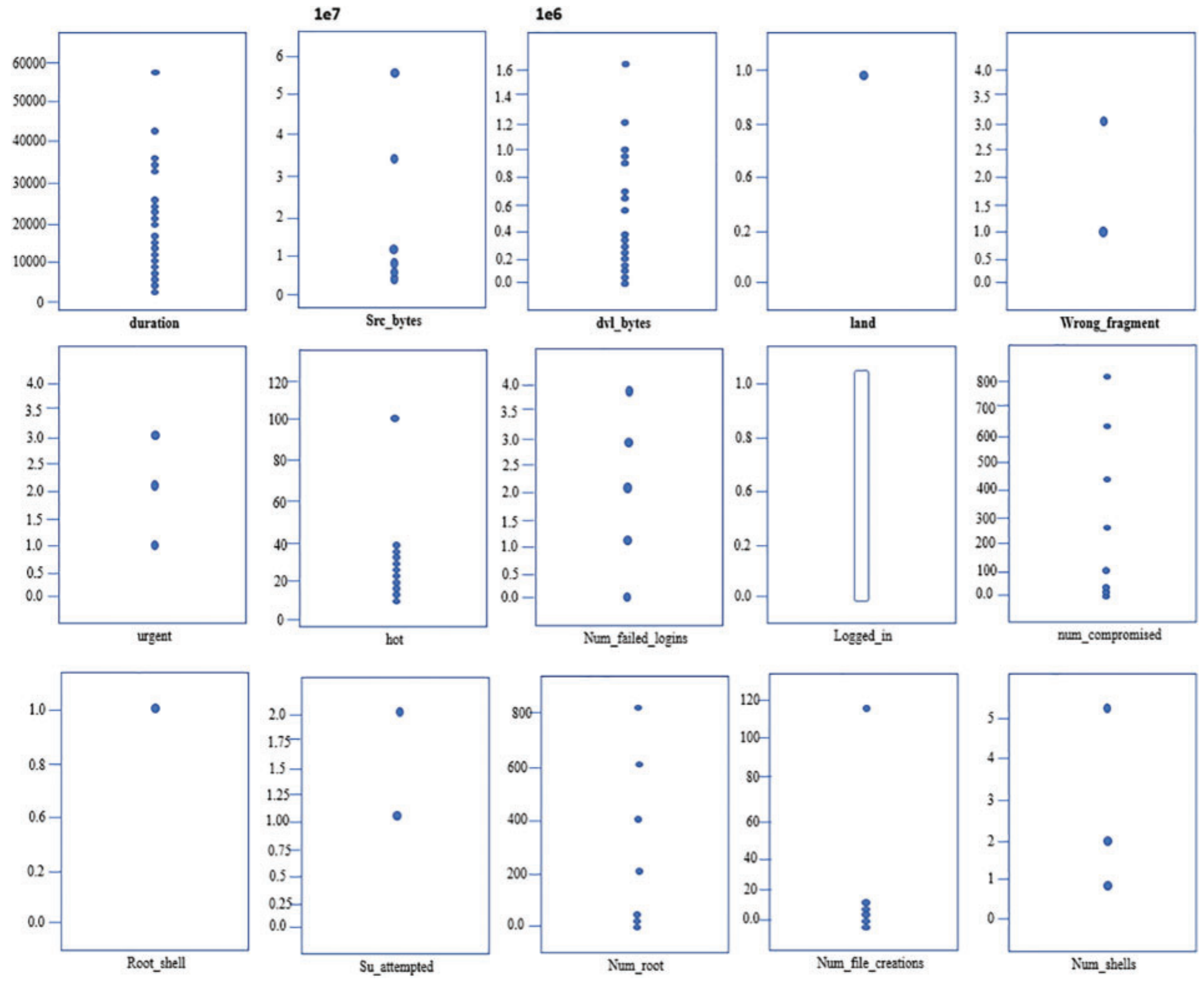

Figure 5: Test data outlier

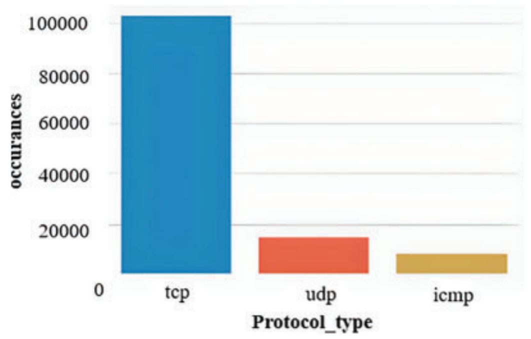

Figure 6: Categorical feature protocol_type *vs*. occurrences

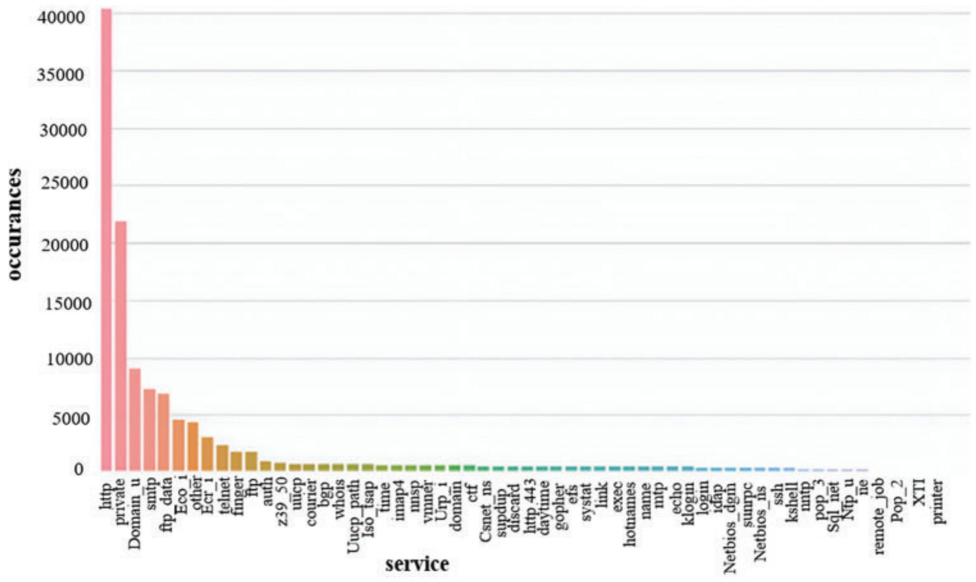

Figure 7: Categorical feature 'service' *vs*. occurrences

Check out the dataset's current open flag count. Fig. 8 depicts categorical feature flags *vs.* occurrences.

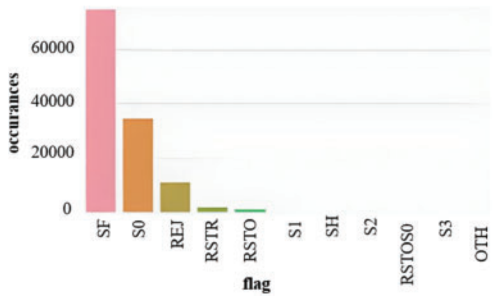

Figure 8: Categorical feature 'flag' *vs*. occurrences

Fig. 9 depicts a graph of class *vs*. occurrences with several class categories, such as normal, Neptune, satan, portsweep, smurf, and nmap.

The data was separated into four main attack classes; the next step is to visually represent the number of attacks present in each class and contrast it with normal, as shown in Fig. 10.

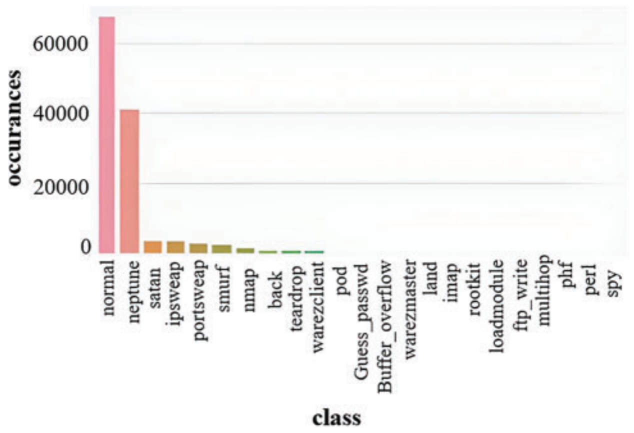

Figure 9: Categorical feature 'class' *vs*. occurrences

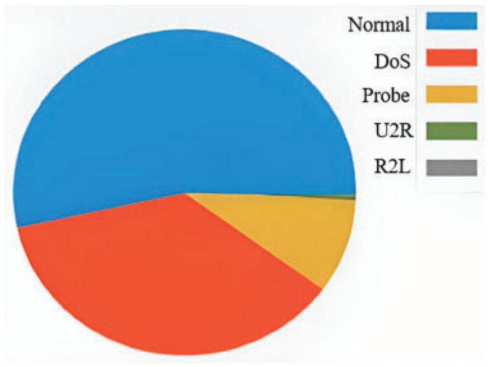

Figure 10: Attack type *vs.* normal

4 Methodology

The NSL-KDD dataset, which fixes several issues with the KDD Cup'99 dataset, has replaced the KDD-CUP dataset. Four assault categories in the NSL-KDD dataset exhibit aberrant data, while one normal category demonstrates that the associated cases are typical.

The flow chart below, displayed in Fig. 11, suggests the steps taken during this investigation. Data identification comes first, then pattern recognition, and finally data pre-processing. Next, the training set is used to choose the algorithm. Finally, if the condition is true, the classifiers are examined; otherwise, the previous procedures are employed again.

4.1 Data Pre-Processing

The NSL-KDD data set has undergone several pre-processing procedures to improve its suitability for use in IDS. These methods include reducing unnecessary features, converting continuous values into discrete ones, and removing superfluous entries. The data set also includes a set of labeled examples of both normal and abnormal network connections. These examples can be used to test and train IDS algorithms. Among the 38 quantitative parameters in the NSL-KDD dataset are three non-numerical features. The RNN-input IDS should be a number matrix; thus, we must convert the non-numeric attributes into numeric values. Three possibilities were provided by the functionality “protocol”: types of features service with 70 various benefits, and features “flags” with 11 distinct advantages. After conversion, the 41-dimensional characteristics from the Americanized NSL-KDD data set became 122-dimensional features.

4.2 Principal Component Analysis (PCA)

Choosing the core features that best capture the comprehension of the material is a quantitative technique for converting frequent occurrences into low-dimensional data. The traits are selected based on the variation in the output they produce. The first main element that offers the most diversity is the feature. The second input parameter is the characteristic that explains the second-highest degree of variance. An illustration of PCA is shown in Fig. 12 where we can see the variation in data. It is crucial to emphasize that the core components were not connected in any way.

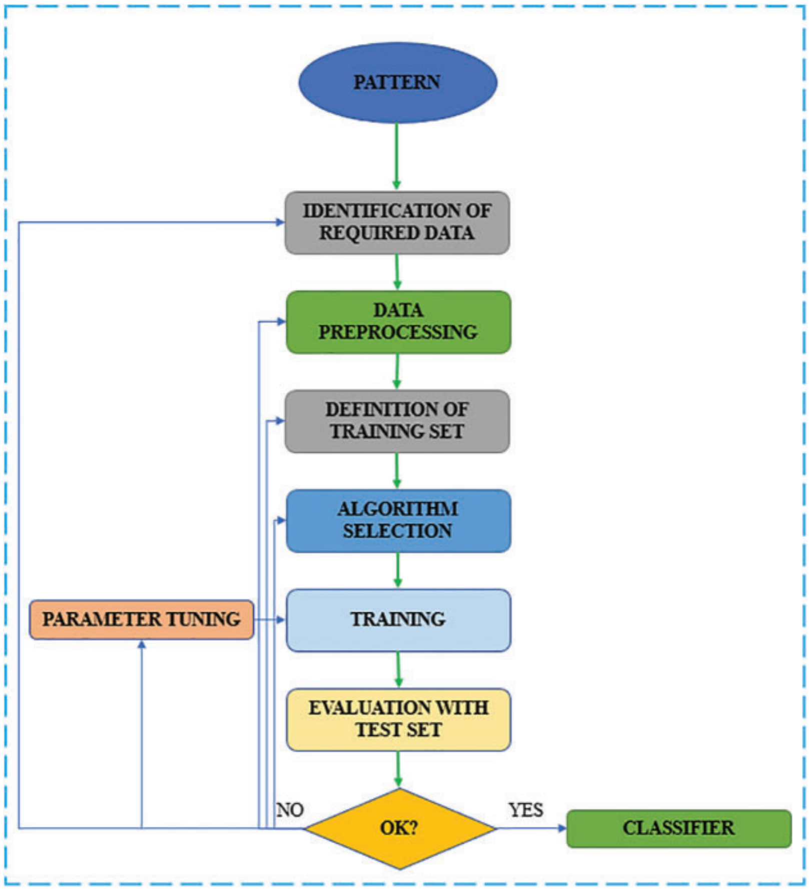

Figure 11: Our research methodology

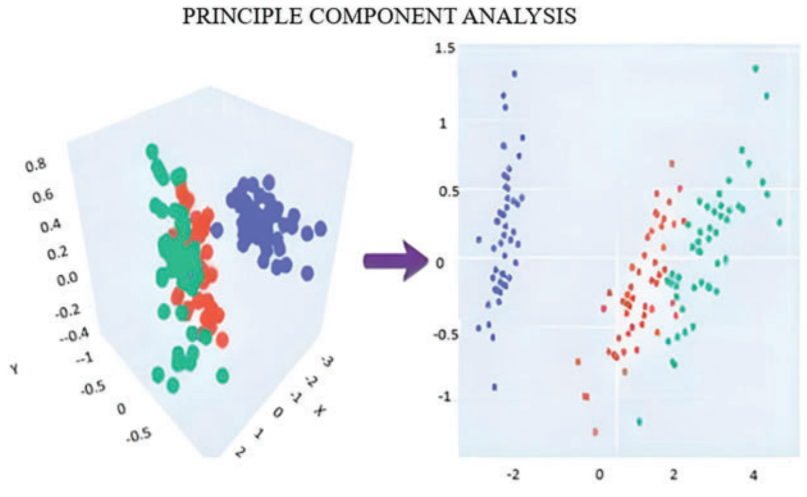

Figure 12: PCA

4.3 Benefits of Principle Component Analysis (PCA)

There are two main advantages of PCA data processing.

- The training time for the procedures was drastically reduced with fewer elements.
- Only occasionally is it practical to analyze data in unprecedented detail. Consider a dataset with 100 characteristics, for instance.

A 100 (100−1)2 = 4950 histogram would be required to depict the data. Unfortunately, this type of data analysis is not practical in real life. We acquired 122 original features and 20 reduced features after PCA was used.

By applying the PCA approach to reduce its dimensionality, the NSL-KDD data collection could be considered more accessible for IDS to process. To implement PCA on the NSL-KDD data set, follow these steps:

- Data should be normalized so that the mean and variance are zeroes because PCA is sensitive to scale.
- The covariance matrix, which is a square matrix, describes the correlations between the various characteristics of the data set.
- Calculate the covariance matrix's eigenvalues and eigenvectors.
- The highest directions of variance in the data set are called eigenvectors, and the corresponding variances are called eigenvalues.
- Since the goal of PCA is to reduce the dimensionality of the data set, a selection of the eigenvectors with the highest eigenvalues is made. Data projection onto the new eigenvectors: The data set is then projected onto the chosen eigenvectors to be converted into a lower-dimensional space.

The calculation cost can be reduced by using the new low-dimensional data to train IDS. However, be advised that depending on the algorithm and demands of the DSI, the PCA implementation on the NSL-KDD data set may provide varied results. Therefore, comparisons between a model's performance before and after the PCA application are necessary.

Before applying PCA, the NSL-KDD data set has a high degree of dimension and many features. IDS may find it challenging to process and comprehend the data.

A lot of the characteristics could also be highly linked, which might result in overfitting and subpar generalization performance. The dimensionality of the data collection is decreased after PCA by choosing a subset of the most crucial features. In addition to lowering the risk of overfitting, this can facilitate the processing and comprehension of the data by IDS. The performance of IDS can be enhanced using PCA by removing noise and redundancy from the data stream. It is crucial to remember that PCA is a linear technique and cannot detect non-linear relationships between the data.

Additionally, it will not take the class labels into account when choosing the features; instead, it will choose the features that account for the most variation in the data, regardless of the class label. Therefore, it makes sense to assess the model's performance before and after using PCA, then contrast the outcomes. Other dimensionality reduction methods, like an autoencoder, LLE, t-SNE, etc., are also worth considering. PCA aims to minimize the size of the dataset while preserving as much variability as possible from the original dataset. It is a technique for identifying data-based patterns and highlighting the differences and similarities across the data. Fig. 13 illustrates the use of PCA in the training set, which shortens the training period.

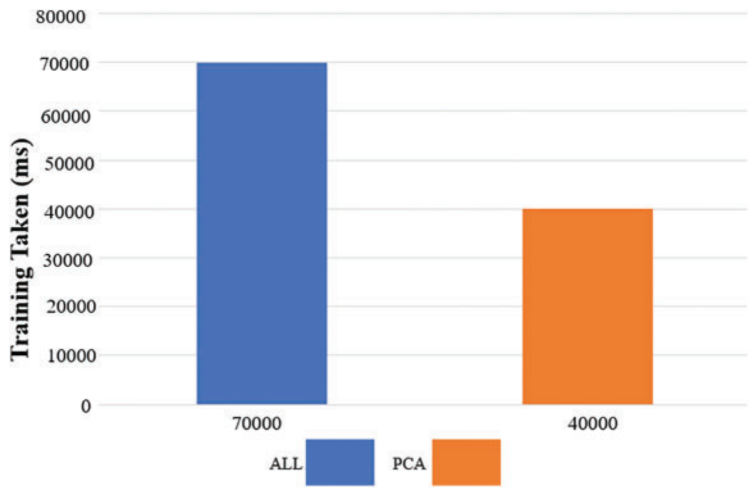

Figure 13: Before and after PCA

4.4 Artificial Neural Network (ANN) Modeling

Using algorithms and machine learning techniques, an NN is a computer program that mimics the functions of the human brain. NN features more robust computer techniques, real-time processing capabilities, and more significant data handling capabilities when compared to conventional computers. Neuronal networks are also known as ANNs. An input layer, one or more hidden layers, or an output layer are all covered by node layers in the design of a brain program. Nodes are "artificial neuron" clusters with a specific weight and threshold. A node turns on and sends data into the network's next layer when its output rises above certain thresholds, as depicted in Fig. 14. Data is not transmitted to the next network layer if the node's limit is not reached.

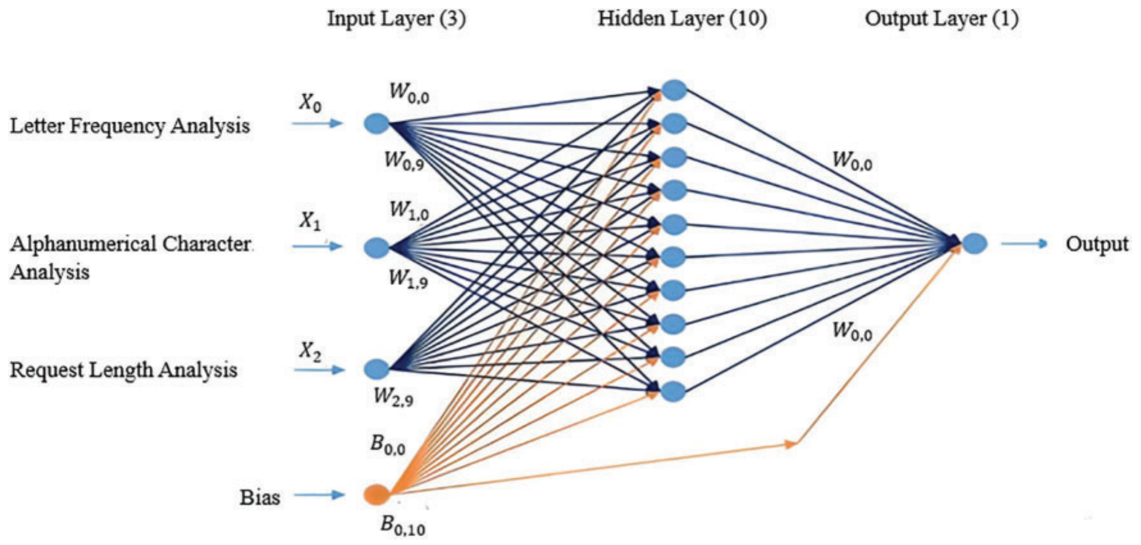

Figure 14: ANN structure

Additionally, these adaptive computations offer highly sophisticated design tools that allow us to quickly categorize and analyze cluster data once they have been optimized for efficiency. Processes like NLP or image identification may be finished in minutes rather than hours compared to manual verification by human experts. Google's algorithm is built on one of the most well-known ANNs.

4.5 Implementation of Artificial Neural Network (ANN)

4.5.1 Data Splitting

The NSL-KDD data set must be separated into separate files to be ready for IDS. Therefore, the training, evaluation, and test sets are divided from the entire data set during the data-splitting procedure.

- Training set: This set serves as the training data for the IDS. Usually, 70% and 80% of the data set is used for training.
- Validation set: Using this set, the IDS's parameters are changed during the training phase. This set typically has a 10%–15% reduction in size compared to the training set.
- Test set: Using this set, the IDS's performance shall be evaluated upon training. Therefore, it is essential to have a test set distinct from the training and validation sets. In addition, the test set needs to be sizable (e.g., 10%–15%) to accurately measure the IDS's effectiveness.

The impact of the data splitting ratio can change depending on the methodology and requirements for the IDS, and it is crucial to remember. For example, various data-splitting ratios might be used to evaluate the model's performance. To guarantee that the sample represents the complete data set, the data should be divided randomly. Keeping a distinct set of data, such as an evaluation set, that is only used for evaluating the performance of the final model rather than being utilized for training or testing the IDS is also a good practice.

4.5.2 *Train Data*

The four kinds of train data are as follows, as indicated in Table 3.

Table 3: Train data

Dos	391458
Normal	97278
Probe	4107
R2L	1126
U2R	52
Name: target, dtype: int64	

We require two sets of data to train and test the data. The data description after splitting is shown in Fig. 15. The figure on the right depicts the Attack *vs*. outcome data sets. If 47% represents the Attack and 53% is the usual outcome type. The procedure type *vs*. outcome is shown, on the other hand, in the figure on the left. Where the percentages of TCP, icmp, and udp are 82%, 7%, and 12%, respectively.

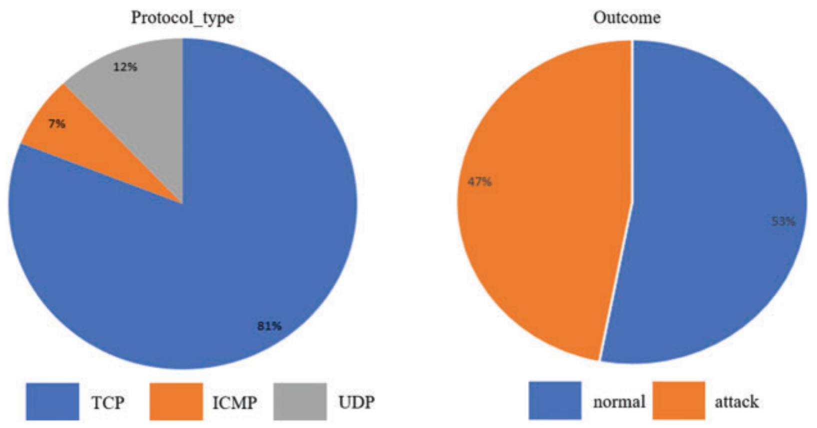

Figure 15: Data after splitting

4.5.3 *Artificial Neural Network (ANN) Model Implementation*

The ANN is created using the open-source software package called Keras. Each buried layer's related neurons are shown in Table 4. The ReLU function of the ANN model is used to activate both the input layer and each hidden layer. Eq. (1) displays the activation function of ReLU. This simple linear function directly generates the input whenever the input is positive; else, the output is zero. A collection of active nodes is termed a "rectified activation functions unit" using this function.

$$ReLu\,(x) = max(0, x) \tag{1}$$

The sigmoid approach was used to activate the output layer; however, it may not map any real benefits to the area [30]. On the other hand, it precisely and thoroughly translates the ANN network's output. The sigmoid function equation is shown in Eq. (2).

$$Sigmoid\,(x) = \frac{1}{1 + e^{-x}} \tag{2}$$

Table 4: Ann model summary

Layer (type)	Output shape	Param #
Dense (dense)	(None, 64)	7872
Dropout (dropout)	(None, 64)	0
Dense 1 (dense)	(None, 128)	8320
Dropout 1 (dropout)	(None, 128)	0
Dense 2 (dense)	(None, 512)	66048
Dropout 2 (dropout)	(None, 512)	0
Dense 3 (dense)	(None, 128)	65664
Dropout 3 (dropout)	(None, 128)	0
Dense 4 (dense)	(None, 1)	129
Total params:148,033		
Trainable params:148,033		
Non-trainable: 0		

Due to the possibility of slope inflation or disappearance, the weights and biases are essential. Gradients increase in size when the starting point is too large, while they decrease in size when it is too small. Therefore, replies should have a mean of zero and constant variation overall levels to avoid the previously discussed problem. Therefore, the layer activated by the RELU function's weights was started using the uniform initializer. The methods used for weight initialization were also utilized to generate the TensorFlow-based Keras object, which is significant. Table 4 shows how bias activation on all levels was accomplished using zero initializers. The ANN's optimization algorithm was Adam, which had a detection rate of 0.001. The loss function used was binarized cross-entropy. It can determine the loss of the model and then adjust the ANN weights to lessen the loss relative to the external. It benefits binary classification problems with target values ranging from 0 to 1.

After the ANN model had been trained using 100 activation epochs, its performance on untrained data was evaluated using the cross-validation technique. If the model is over or under-fitting, it is also feasible to assess the training and cross-validation performance results. The "early pause" method was employed to lessen overfitting. Hyperparameter tuning is required to improve the performance of the ML model. The Keras Tuner module was also used to change the learning rate, normalization factor, number of neurons per layer, and the number of hidden layers.

We worked on the model_plot.png in Keras to plot our model for visualization in as shown in Fig. 16.

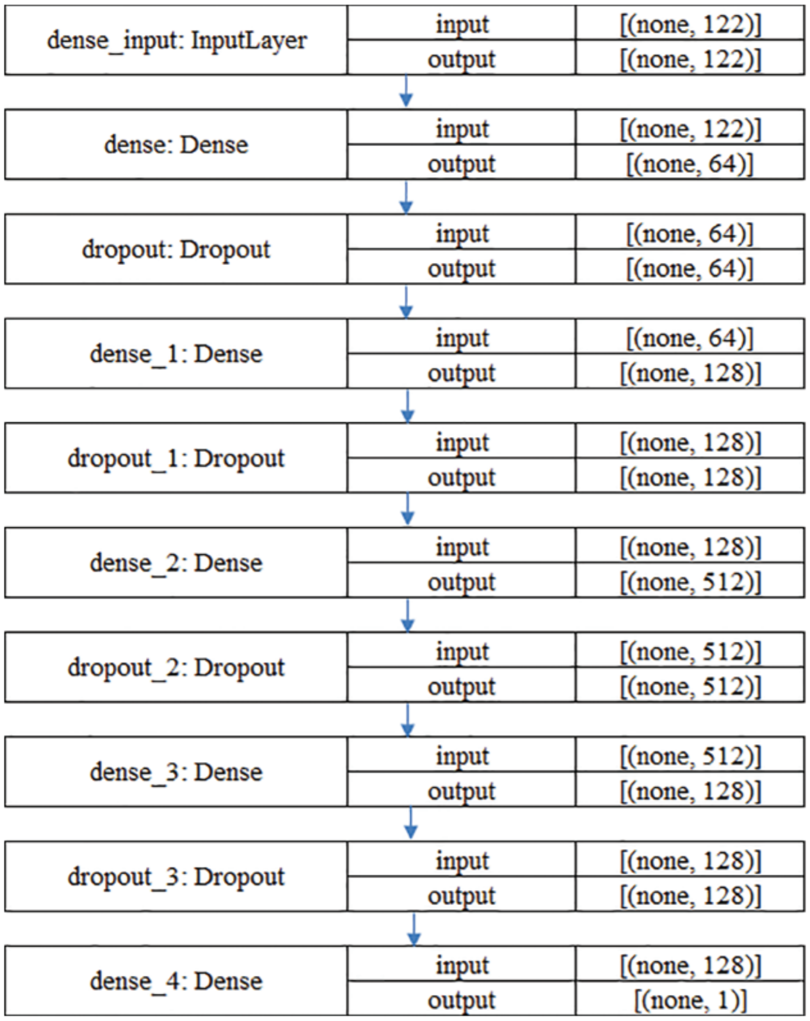

Figure 16: Model architecture

4.6 Computational Complexity of the Proposed Model

Analyzing the computational complexity of any proposed model is crucial to determining its viability and applicability. We explore the computational complexities of the implementation of our suggested intrusion detection model.

Our model makes use of a variety of methods, including ANN for pattern recognition and PCA for dimensionality reduction. The use of PCA helps to decrease the number of features, which may speed up the training process. To avoid overfitting, the training of the ANN model is additionally optimized using strategies like dropout layers. The computational load during both the training and inference phases can be influenced by the model architecture complexity as well as the number of layers and neurons. The accuracy of a model may increase with more complex architectures, but the computational demands may increase as well, making the model more laborious to train and use. Thus, it is critical to strike a balance between model complexity and computational efficiency.

Given the model's important role, we focus on the ANN section in this analysis. The following presumptions will serve as the foundation for our analysis: The input data's sample size is denoted by N. Each sample contains a certain number of features; M. L stands for the neural network's hidden layer count. K denotes how many neurons are present in each hidden layer. O is an abbreviation for the number of output neurons appropriate for binary classification. E stands for the number of training epochs. The weighted sum operation that each neuron in a layer performs, followed by an activation function like ReLU or sigmoid, is a crucial step in the forward pass. The entire layer is subjected to this operation for every neuron. While the space complexity corresponding to intermediate activations in a layer is roughly $O(N * K)$, the time complexity for this forward pass in a single layer is estimated to be $O(N * M * K)$. The computation of gradients and subsequent weight updates for each neuron and layer is part of the backward pass, which includes the backpropagation procedure. The time complexity for this backward pass can be roughly $O(N * L * K * E)$ when L hidden layers and E training epochs are taken into account. The space complexity, tied to the storage of gradients and other temporary variables, is estimated as $O(N * K * L)$.

The overall time complexity of the neural network can be roughly calculated as $O(N * M * K * L + N * L * K * E)$ by adding the complexity of the forward and backward passes. The total space complexity is approximately $O(N * K * L)$ and includes the storage needs for various variables and activations.

5 Results

In this section, we evaluate the fusion of ANN classifier decisions for binary attack detection and present the results on our dataset. The results are then compared to those obtained using a single decision class. We also assessed the efficacy of multiple decision classifiers in network attack analysis. We employ a probabilistic model based on the ANN method for our experiments. As a baseline source, we train our model with pre-processed NSL-KDD samples. We incorporate decision-class features into ANN in each experiment by modifying the classifier decision techniques or merging other selected components that can be flagged and serviced. This paper compares and contrasts several neural network-based DL methods for IDS. We used a classifier in this study to determine whether network traffic intrusions in the NSL-KDD dataset were honest or dishonest.

5.1 Performance Matrices

The trained model is being evaluated using the metrics listed below.

5.1.1 Accuracy

Accuracy measures how frequently the classifier predicts correctly by dividing the total number of guesses by the number of correct predictions.

$$Accuracy = \frac{TN + TP}{TN + FP + TP + FN} \tag{3}$$

An equation measuring accuracy in this situation is Eq (3), which measures the proportion of correctly classified data instances to all other data instances. If the dataset is unequal, using accuracy as a metric might not be acceptable (both negative and positive classes have different numbers of data instances).

5.1.2 Precision

The fraction of expected positives that turn out to be positive is known as precision.

$$Precision = \frac{TP}{TP + FP} \tag{4}$$

Eq. (4) illustrates the accuracy model. Preferably, a good classifier should have a precision of 1 (high).

When TP = TP + FP, or when the numerator and denominator are equal, precision becomes 1, indicating that FP is zero. As FP increases, the accuracy value decreases since the denominator value is greater than the numerator.

5.1.3 Recall

The percentage of properly recognized true positives

$$Recall = \frac{TP}{TP + FN} \tag{5}$$

Eq. (5) depicts the recall equation, where recall for a competent classifier must optimally be 1 (high). The recall becomes one once the numerator and denominator are the same, as in TP = TP + FN, which indicates that FN is zero. The recall value decreases as FN increases because the denominator value exceeds the numerator.

5.1.4 F1 Score

The average of recall and precision harmonically

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{6}$$

In Eq. (6), the F1 score equation is displayed. The F1 score is 1 when precision and recall are both 1. The F1 score can only increase when precision and memory are both excellent. The F1 score, the harmonic mean of recall and precision, is a more helpful indicator than accuracy.

5.2 Experimental Results

When the method's efficacy was assessed, the ANN performed the best, with a maximum detection accuracy of 97.5%, as shown in Table 5. Because the NSL-KDD dataset focuses on attack detection and combined classifiers cannot detect every network connection attack, the gain is small.

Table 5: Matrix measure

Model	Accuracy	F1	Precision	Recall
ANN	97.5%	95.7%	99%	96.7%

Fig. 17's graph shows the training loss, which assesses how well the theory predicts the training data, and the validation loss, which shows how well the model fits new data. Overall, at 4 Epoh, we reduced our validation loss.

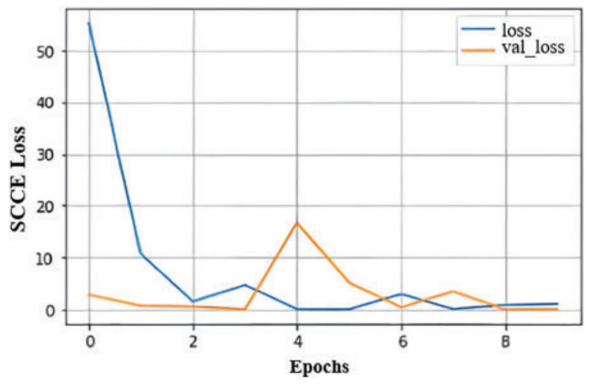

Figure 17: Loss *vs*. validation loss graph

Fig. 18 demonstrates that the training accuracy is smaller than the validation accuracy. Additionally, it can suggest that the model is underfitting. Underfitting happens when the algorithm inaccurately represents the training data and commits many mistakes.

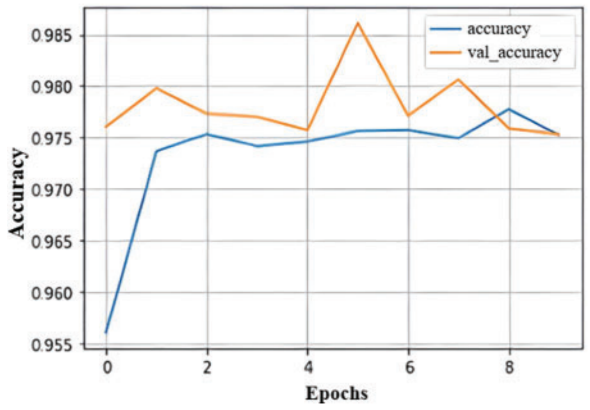

Figure 18: Accuracy *vs*. validation accuracy graph

Fig. 19 shows our model's confusion matrix. Any IDS's efficiency can be expressed in terms of True Negatives (TN), False Negatives (FN), Positives (TP), and False Positives (FP) (FP). When an intrusion is detected, it is classified as TP in the RNN-IDS. False classification regarding network intrusion is labeled as FP. Similarly, TN is assigned when an alert should not be generated without a real-time intrusion. While FN denotes a real-time scenario in which the network is intruded upon, RNN-IDS incorrectly labels it as non-intrusion.

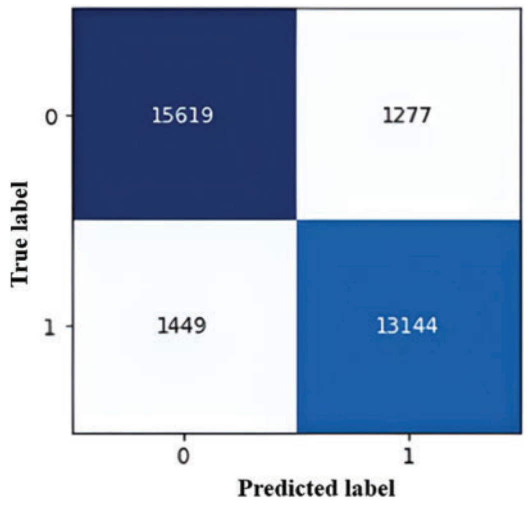

Figure 19: Confusion matrix

5.3 Comparative Analysis

A thorough comparison of various intrusion detection techniques is given in Table 6, which rates their effectiveness using important parameters like accuracy, F1 score, precision, and recall. Each technique, developed by different researchers, makes use of unique methods and models to find network intrusions. K-Means, RF, and DL were combined in [32] to produce results with an accuracy of 85.00%, an F1 score of 86.00%, a precision of 90.00%, and a recall of 87.00%. LSTM is used by [33], which achieves an accuracy of 84.25%, an F1 score of 84.20%, a precision of 84.20%, and a recall of 83.00%. Using a DNN, reference [34] achieved the highest accuracy of 96.00% and an F1 score of 96.20% for binary classification. However, the precision and recall values are not available. With an accuracy of 82.00%, an F1 score of 82.50%, a precision of 85.00%, and a recall of 85.00%, reference [35] introduced the DSSTE method. Reference [36] applied an NB approach and achieved a high accuracy of 97.14%, a stellar F1 score of 97.94%, and a precision of 96.72%. There is no mention of a recall. KPCA is used by [37] and it stands out for its impressive accuracy of 99.00%. The F1 score, on the other hand, is 90.00%, the precision is 95.00%, and the recall is 87.00%. The "Ours" model uses an ANN for binary classification and achieves accuracy of 97.50%, which is comparable to the best techniques on the list. Furthermore, this model outperforms most of the compared methods in terms of precision (99.00%) and recall (96.70%).

Table 6: Performance comparison

Author	Method	Accuracy	F1	Precision	Recall
[32]	K-Means+ Random Forest (RF) and Deep Learning (DL)	85.00%	86.00%	90.00%	87.00%
[33]	Long Short-Term Memory (LSTM)	84.25%	84.20%	84.20%	83.00%
[34]	Deep Neural Network (DNN) binary	96.00%	96.20%	NA	NA
[35]	DSSTE method	82.00%	82.50%	85.00%	85.00%

(Continued)

Table 6 (continued)

Author	Method	Accuracy	F1	Precision	Recall
[36]	Naive Bayes (NB)	97.14%	97.94%	96.72%	NA
[37]	Kernel-based principal component analysis (KPCA)	99.00%	90.00%	95.00%	87.00%
Ours	Artificial Neural Network (ANN) binary	97.50%	95.70%	99.00%	96.70%

The method [34] that used a DNN for binary classification and attained the highest accuracy of 96.00% among the ones mentioned above. It is crucial to remember that this model's precision and recall values were not provided. The NB approach was used by [36] in contrast, and it resulted in an impressive accuracy of 97.14% as well as high F1 scores of 97.94% and precision of 96.72%. However, no recall information was given. Due to its high Precision and Recall scores, our model is able to effectively reduce false positives (misclassification of regular occurrences as intrusions) and false negatives (failure to detect actual intrusions). To maintain network security and lessen the impact of potential threats, balanced performance is crucial. The comparison table shows unequivocally that our suggested ANN-based model outperforms the listed approaches in terms of Accuracy, F1 score, Precision, and Recall, emphasizing its effectiveness in intrusion detection and its potential to offer a more trustworthy defense against changing network threats.

It is clear that the suggested ANN-based model achieves high accuracy and maintains a remarkable balance between precision and recall in addition to achieving high accuracy. In order to accurately identify real intrusions while reducing false alarms, this balance is essential in intrusion detection. As a result, the proposed model performs better overall than a number of current algorithms and offers a reliable solution for intrusion detection tasks.

Our ANN model's outstanding performance can be attributed to a thorough methodology that includes data pre-processing, PCA, exact ANN implementation, hyperparameter tuning, and balanced data splitting. Our data pre-processing included feature reduction, discretizing continuous data, and removing duplicate entries. The incorporation of PCA facilitated dimensionality reduction while safeguarding important data, improving training effectiveness, and lowering the risk of overfitting. The ANN model was painstakingly designed, utilizing activation functions, multiple layers and neurons, and dropout layers for improved generalization. The model was further optimized in Section 4.5.3 through effective hyperparameter tuning. Last but not least, a solid foundation for training, validation, and testing was guaranteed by our balanced data splitting in Section 4.5.1. Altogether, these elements collectively empower our model to excel in intrusion detection, surpassing other methodologies in performance.

6 Discussion

Network systems are vulnerable to several threats because of the enormous growth in internet-based traffic. Through its use of time and resources, intrusive traffic hinders the efficient operation of network infrastructure. Productivity is increased through adequate intrusion incident protection, detection, and mitigation techniques. IDS is one of the essential elements of secure network traffic. To comprehensively monitor intrusive network traffic, an IDS system might be host-based or network-based. The effectiveness of automated anomaly traffic detection methods is increasing with time. By testing with several machine learning techniques, this study seeks to identify the optimal classifier that, from the NSL-KDD dataset, can identify anomalous traffic with a high level of accuracy and a low error rate.

Due to the massive increase in internet traffic, modern networks are exposed to various dangers. Intrusive traffic prevents the effective operation of network infrastructure by consuming time and resources. IDS is one of the techniques that aid in assessing system security because it alerts users when an intrusion is discovered. In addition, networked communication anomalies and breaches must be detected using ML. Regularization, one of the fundamental ideas in ML model training, is vital to the success of many successful ANN models since regularisation is brought about throughout the model's training.

Moreover, the curve in Fig. 17 depicts the training loss, which measures how well the theory forecasts training data, and the validation loss, which depicts how well the model fits new data. Overall, we decreased our validation loss at 4 Epoh. The training loss is less than the VAL accuracy, as shown in Fig. 18. However, it also implied that the model does not fit properly. Underfitting occurs when the algorithm depicts the training data incorrectly and makes numerous errors.

Finally, the ANN appears to have the best overall performance after selecting the top seven features using techniques including K-Means+, RF and DL, LSTM, DNN, the DSSTE approach, and the ANN. The F1 score has an accuracy of 86%, a precision of 90%, and a recall of 100%, compared to the 85% accuracy of K-Means, RF, and DL. For the LSTM technique, the values for accuracy, F1 score, precision, and recall are 84.25%, 84.2%, 84.2%, and 83%, respectively. The DNN binary technique has a 96.2% accuracy rate and an F1 score. The DSSTE method has values for accuracy, F1 score, precision, and recall which are all 85%. The F1 score is 82.5%, recall is 85%, and accuracy is 82%. Finally, our ANN method performs significantly better than all other methods. While accuracy scores 97.5% and has an F1 score of 95.7%, precision scores 99% and has a recall value of 96.7%.

ANNs are widely used in intrusion detection, and our study has significantly advanced this field. However, some restrictions must be recognized. Our main source of data is the NSL-KDD dataset, which, while addressing some KDD Cup'99 issues, might not accurately represent the complexity of real-world network traffic, which would have an impact on generalizability. PCA that we used to select our features may have missed subtle relationships that would have improved detection precision. Our ANN model's optimized architecture and complexity may change depending on the configuration. A biased dataset distribution could indicate over-, under-, or synthetic data techniques by skewing the model's predictions. Although useful for binary and multiclass settings on the NSL-KDD dataset, it is still unknown whether the model can be applied to other datasets, fresh attacks, and different network configurations. Scalability and performance may be impacted by hardware constraints and tightly controlled experimental environments. Last but not least, further research is needed into real-world deployment issues like changing attack strategies and network dynamics.

Although the proposed intrusion detection scheme has promise, there are legitimate concerns about its practical application. Assuring data diversity and quality for accurate model performance, protecting against adversarial attacks that could fool the model, accounting for concept drift to prevent degradation over time, and optimizing feature engineering to capture contemporary attack nuances without introducing bias are challenges. Additional issues include model overfitting, resource-intensive computations, responding to new attacks, and maintaining interpretability. Effectiveness is also impacted by deployment difficulties, data security and privacy concerns, and the trade-off between false positives and false negatives. A strong evaluation framework that includes real-world testing, ongoing adaptation, a variety of datasets, security precautions, and expert collaboration is essential to addressing these threats and ensuring the validity of the suggested scheme in practical situations.

7 Conclusion

In the field of IDS, machine learning techniques have attracted a lot of interest. Using multiple algorithms together frequently produces better outcomes than using just one. This idea emphasizes how crucial algorithmic synergy is. While many researchers concentrate on the classification component of IDS, which is primarily intended to identify known intrusion attacks, the detection of anomalous intrusions, which includes novel or modified attacks, presents a challenge. Therefore, incorporating clustering algorithms into IDS development in the future holds promise for enhancing its robustness. The evaluation is conducted using the NSL-KDD dataset, and it examines the efficacy of these combined strategies in binary and multiclass classification settings. It is noteworthy that our method departs from the Deep Learning-based feature selection path, choosing instead to use NSL-KDD in the creation of an ANN-driven IDS. Our research operationalized an ANN-based IDS through rigorous experimentation, training and testing it against five different attack categories and a binary classification scenario (normal or attack). It is noteworthy how imbalanced training sets were handled specifically for the R2L and U2R categories, selectively incorporating some patterns from other classes. The conclusion is clear: with a recall rate of 96.7% and a precision score of 99%, our ANN method significantly outperforms competing strategies. The trajectory shown in this study signals a promising path for robust intrusion detection in dynamic and constantly changing network environments as we navigate the complexities of IDS advancement.

Future research projects might examine the incorporation of sophisticated clustering algorithms to increase the IDS's resistance to new and sophisticated intrusion techniques. Investigations into the adaptation of ANNs with dynamic features and changing attack vectors may also provide new information that can be used to strengthen network security measures. In addition, investigating hybrid models that combine the benefits of various machine learning techniques may help create IDS solutions that are more precise and adaptable in the face of changing cyber threats.

Acknowledgement: The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research.

Funding Statement: The authors extend their appreciation to the Deputyship for Research & Innovation, Ministry of Education in Saudi Arabia for funding this research (IFKSURC-1-7109).

Author Contributions: Conceptualization, M.Z. and S.A.; methodology, M.Z. and S.A.; software, M.Z.; validation, M.Z. and S.A.; formal analysis, M.Z. and S.A.; investigation, M.Z. and S.A.; resources, S.A.; data curation, M.Z.; writing—original draft preparation, M.Z. and S.A.; writing—review and editing, M.Z. and S.A.; visualization, M.Z. and S.A.; supervision, S.A.; project administration, S.A.; funding acquisition, S.A.

Availability of Data and Materials: The dataset is publicly available and can be given on reasonable request.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. A. Albahar, M. Binsawad, J. Almalki, S. El-etriby and S. Karali, "Improving intrusion detection system using artificial neural network," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, pp. 578–588, 2020.

[2] R. Abdulhammed, M. Faezipour, A. Abuzneid and A. AbuMallouh, "Deep and machine learning approaches for anomaly-based intrusion detection of imbalanced network traffic," *IEEE Sensors Letters*, vol. 3, no. 1, pp. 1–14, 2019.

[3] A. Verma and V. Ranga, "On the evaluation of network intrusion detection systems: Statistical analysis of CIDDS-001 dataset using machine learning techniques," *Pertanika Journal of Science & Technology*, vol. 26, no. 1, pp. 1307–1332, 2018.

[4] A. Verma and V. Ranga, "Statistical analysis of CIDDS-001 dataset for network intrusion detection systems using distance-based machine learning," *Procedia Computer Science*, vol. 125, no. 2, pp. 709–716, 2018.

[5] A. Tama and K. H. Rhee, "An in-depth experimental study of anomaly detection using gradient boosted machine," *Neural Computing and Applications*, vol. 31, no. 4, pp. 955–965, 2019.

[6] A. Tama, M. Comuzzi and K. H. Rhee, "TSE-IDS: A two-stage classifier ensemble for an intelligent anomaly-based intrusion detection system," *IEEE Access*, vol. 7, pp. 94497–94507, 2019.

[7] A. L. Maglaras, M. A. Ferrag, M. Derdour and H. Janicke, "A novel hierarchical intrusion detection system based on decision tree and rules-based models," in *Proc. of SecRIoT 2019, 1st Int. Workshop on Security and Reliability of IoT Systems*, Santorini Island, Greece, 2019.

[8] S. Solanki, C. Gupta and K. Rai, "A survey on machine learning based intrusion detection system on NSL-KDD dataset," *International Journal of Computer Applications*, vol. 176, no. 30, pp. 36–39, 2020.

[9] J. J. Tanimu, M. Hamada, P. Robert and A. Mahendran, "Network intrusion detection system using deep learning method with KDD Cup'99 Dataset," *2022 IEEE 15th Int. Symp. on Embedded Multicore/Many-Core Systems-on-Chip (MCSoC)*, Penang, Malaysia, pp. 251–255, 2022.

[10] M. Hafeez, A. Y. Ding and S. Tarkoma, "IoT-KEEPER: Detecting malicious IoT network activity using online traffic analysis at the edge," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 45–59, 2020.

[11] S. Hanif, T. Ilyas and M. Zeeshan, "Intrusion detection in IoT using artificial neural networks on UNSW-15 Dataset," in *Proc. of 2019 IEEE 16th Int. Conf. on Smart Cities: Improving Quality of Life Using ICT & IoT and AI (HONET-ICT)*, Charlotte, NC, USA, pp. 152–156, 2019.

[12] K. A. Taher, B. M. Y. Jisan and M. M. Rahman, "Network intrusion detection using supervised machine learning technique with feature selection," in *Proc. of 2019 Int. Conf. on Robotics, Electrical and Signal Processing Techniques (ICREST)*, Dhaka, Bangladesh, pp. 643–646, 2019.

[13] N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41–50, 2018.

[14] P. Peter and G. Louppe, "Gradient boosted regression trees in scikit-learn," in *PyData 2014*, London, UK, 2014.

[15] P. Fabian, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion *et al.,* "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[16] K. Sydney and Y. Sun, "Performance analysis of intrusion detection systems using a feature selection method on the UNSW-NB15 dataset," *Journal of Big Data*, vol. 7, no. 1, pp. 1–20, 2020.

[17] B. F. Wu and C. H. Lin, "Adaptive feature mapping for customizing deep learning-based facial expression recognition model," *IEEE Access*, vol. 6, no. 1, pp. 12451–12461, 2018.

[18] C. Yin, Y. Zhu, J. Fei and X. He, "A deep learning approach for intrusion detection using recurrent neural networks," *IEEE Access*, vol. 5, no. 21, pp. 954–21, 961, 2017.

[19] Z. Ahmad, A. S. Khan, C. W. Shiang, J. Abdullah and F. Ahmad, "Network intrusion detection system: A systematic study of machine learning and deep learning approaches," *Transactions on Emerging Telecommunications Technologies*, vol. 32, no. 1, pp. 1–29, 2021.

[20] R. Vinayakumar, M. Alazab, K. P. Soman, P. Poornachandran, A. Al-Nemrat *et al.,* "Deep learning approach for intelligent intrusion detection system," *IEEE Access*, vol. 7, no. 1, pp. 41525–41550, 2019.

[21] C. Zina, A. Erbad, R. Hamila, A. Mohamed, M. Guizani *et al.,* "TIDCS: A dynamic intrusion detection and classification system based feature selection," *IEEE Access*, vol. 8, no. 2, pp. 95864–95877, 2020.

[22] M. H. Ali, B. A. D. A. Mohammed, M. A. B. Ismail and M. F. Zolkipli, "A new intrusion detection system based on fast learning network and particle swarm optimization," *IEEE Access*, vol. 6, no. 2018, pp. 20255–20261, 2018.

[23] S. Arindam, H. S. Sharma and M. M. Singh, "A supervised machine learning-based solution for efficient network intrusion detection using ensemble learning based on hyperparameter optimization," *International Journal of Information Technology*, vol. 15, no. 1, pp. 423–434, 2023.

[24] A. Adeel, H. Larijani and A. Ahmadinia, "Random neural network based novel decision-making framework for optimized and autonomous power control in LTE uplink system," *Physical Communication*, vol. 19, no. 4, pp. 106–117, 2016.

[25] A. F. Agarap, "Deep learning using rectified linear units (relu)," arXiv preprint. arXiv:1803.08375, 2018.

[26] S. Sumaiya, C. Rupa, G. Srivastava and T. R. Gadekallu, "Botnet attack intrusion detection in IoT enabled automated guided vehicles," in *2022 IEEE Int. Conf. on Big Data (Big Data)*, Osaka, Japan, IEEE, pp. 6332–6336, 2022.

[27] S. Gautam, N. Deepa, B. Prabadevi and P. K. Reddy, "An ensemble model for intrusion detection in the Internet of Softwarized Things," in *Adjunct Proc. of the 2021 Int. Conf. on Distributed Computing and Networking*, Nara, Japan, pp. 25–30, 2021.

[28] A. Shaashwat, S. Sarkar, O. Aouedi, G. Yenduri, K. Piamrat *et al.,* "Federated learning for intrusion detection system: Concepts, challenges and future directions," *Computer Communications*, vol. 195, no. 1, pp. 346–361, 2022.

[29] C. Annadurai, I. Nelson, K. N. Devi and R. Manikandan, "Biometric authentication-based intrusion detection using artificial intelligence Internet of Things in smart city," *Energies*, vol. 15, no. 19, pp. 7430–7444, 2022.

[30] S. M. Kasongo, "A deep learning technique for intrusion detection system using a recurrent neural networks based framework," *Computer Communications*, vol. 199, no. 1, pp. 113–125, 2023.

[31] S. M. Kasongo and Y. Sun, "A deep learning method with filter based feature engineering for wireless intrusion detection system," *IEEE Access*, vol. 7, no. 1, pp. 38597–38607, 2019.

[32] C. Liu, Z. Gu and J. Wang, "A hybrid intrusion deection system based on scalable K-Means+ random forest and deep learning," *IEEE Access*, vol. 9, no. 1, pp. 75729–75740, 2021.

[33] T. Su, H. Sun, J. Zhu, S. Wang, Y. Li *et al.,* "Deep learning methods on network intrusion detection using NSL-KDD dataset," *IEEE Access*, vol. 8, no. 1, pp. 29575–29585, 2020.

[34] R. Vinayakumar, R. Chaganti and Mamoun Alazab "Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system," *Computers and Electrical Engineering*, vol. 102, no. 1, pp. 108156, 2022.

[35] P. P. Wagle, "Machine learning-based ensemble network security system," in *Recent Advances in Artificial Intelligence and Data Engineering*. Singapore: Springer, pp. 3–15, 2022.

[36] S. Y. Kayode, A. I. Abiodun, S. Y. Misra, M. K. Holone and R. Colomo-Palacios, "A machine learning-based intrusion detection for detecting internet of things network attacks," *Alexandria Engineering Journal*, vol. 61, no. 12, pp. 9395–9409, 2022.

[37] R. Vinayakumar, R. Chaganti and M. Alazab, "Recurrent deep learning-based feature fusion ensemble meta-classifier approach for intelligent network intrusion detection system," *Computers and Electrical Engineering*, vol. 10, no. 1, pp. 108156–108170, 2022.