Eurasian J. Sci. Eng., 2024, 10(3), 85-96
https://doi.org/10.23918/eajse.v10i3p9

Published by Tishk International University
Available at: https://eajse.tiu.edu.iq/

RESEARCH ARTICLE Open Access

Machine Learning-Driven Intrusion Detection Systems: Reducing False Alarms and Enhancing Accuracy

Safwan Mawlood Hussein[1]*, and Abubakar Muhammad Ashir [1]

[1] Computer Engineering Department, Faculty of Engineering, Tishk International University, Erbil, Iraq.

Article History
Received: 27.01.2025
Revised: 02.03.2025
Accepted: 10.03.2025
Published: 12.03.2025
Communicated by: Dr. Orhan Tug
**Email address:*
safwan.mawlud@tiu.edu.iq
**Corresponding Author*

Abstract: The increasing sophistication of cyber threats presents ongoing challenges for securing modern networks, particularly in addressing the limitations of Intrusion Detection Systems (IDS). Traditional IDS solutions often suffer from high false-positive rates and limited accuracy in detecting novel or unknown attacks, leading to inefficiencies in security management. This paper explores the use of multiple Machine Learning (ML) algorithms to improve IDS performance, focusing on models such as Artificial Neural Networks (ANN), K-Nearest Neighbors (KNN), Decision Trees (DT), Naive Bayes (NB), Logistic Regression (LR), and Support Vector Machines (SVM). The research employs the KDD Cup 1999 dataset, a well-known benchmark for intrusion detection, to evaluate the effectiveness of these models. The study also investigates the role of Principal Component Analysis (PCA) improves model efficiency by reducing the dimensionality of the feature set. Experimental results demonstrate that the integration of ML algorithms significantly improves IDS accuracy while reducing false alarms. This research offers valuable insights into addressing key IDS limitations and provides a comprehensive performance comparison to identify the most suitable model for real-world application.

Keywords: *Intrusion Detection System; Machine Learning; KDD Cup 1999, Hybrid IDS; Anomaly-Based Detection; Signature-Based Detection.*

1. Introduction

In recent years, the increasing complexity of cyber-attacks has underscored the need for more robust and intelligent systems to detect and mitigate threats. The use of technology in everyday activities has made digital systems vulnerable to numerous attack vectors. Consequently, cybersecurity has become a critical concern, with Intrusion Detection Systems (IDS) being one of the most vital tools in identifying and mitigating unauthorized access or abnormal activities in a network. Traditional IDS approaches, while effective, often suffer from high false-positive rates and struggle to detect newer, more sophisticated attack types. Machine learning techniques offer a promising solution to address these challenges, enabling IDS to learn from data patterns and detect anomalies with greater precision and accuracy.

Machine learning-based IDS can be divided into two broad categories: anomaly detection and misuse detection. While misuse detection focuses on recognizing known attack signatures, anomaly detection identifies deviations from normal network behavior, which may indicate an attack. Machine learning algorithms, by learning from vast datasets, can improve both the accuracy and efficiency of IDS systems. However, a key challenge in these systems is balancing high detection rates with minimal false positives to prevent alarm fatigue and ensure timely threat mitigation.

The primary challenge in modern intrusion detection lies in increasing the accuracy of attack detection while lowering the number of false alarms, which often lead to unnecessary administrative interventions. Current signature-based methods depend on predefined attack patterns, rendering them ineffective against zero-day attacks or new variants. To overcome these shortcomings, machine learning methods have been employed. However, different machine learning algorithms exhibit

varying levels of performance in terms of accuracy, precision, recall, and computational complexity, making it crucial to select the most appropriate model for the IDS framework.

The methodology presented in this study aims to integrate a variety of machine learning algorithms, including ANN, K- KNN, DT, NB, LR, and SVM, into the IDS framework to improve its detection capabilities. Furthermore, techniques such as feature scaling, one-hot encoding, and PCA will be applied to the dataset to enhance model performance and ensure that the resulting IDS can efficiently detect attacks with minimal false alarms.

The main objective of this study is to design and evaluate an enhanced IDS framework that leverages various machine learning algorithms. The study focuses on assessing the effectiveness of various machine learning models., such as ANN, KNN, DT, NB, LR, and SVM, in terms of their ability to detect network intrusions an essential goal is to reduce the number of false positives generated by these models while maintaining or improving their detection accuracy.

In addition, this research investigates the role of PCA in improving the performance of machine learning models by reducing the dimensionality of the dataset. The dimensionality reduction is expected to not only enhance computational efficiency but also improve the overall performance of the IDS. Finally, the study aims to perform a comprehensive performance comparison across the different machine learning models, evaluating them based on standard metrics such as precision, recall, accuracy, and the confusion matrix, to determine which model provides the most balanced and effective intrusion detection performance.

2. Literature review

An IDS is an essential component of modern cybersecurity frameworks, designed to detect unauthorized access or malicious activities within a network. Over the years, various approaches have been explored to enhance the performance of IDS, with machine learning emerging as one of the most promising techniques. This chapter provides an in-depth review of existing literature related to IDS, focusing on machine learning algorithms, dimensionality reduction techniques, and evaluation metrics used to optimize detection accuracy while minimizing false positives.

2.1 Traditional Intrusion Detection Systems

Intrusion Detection Systems are essential security mechanisms developed to detect unauthorized access or improper use of system resources within a host or network. IDS works by collecting and analyzing network traffic and generating alerts when suspicious activities are detected, allowing network managers to respond to potential intrusions[1].

Traditional IDS relies on two primary methodologies: signature-based detection and anomaly-based detection. Signature-based IDS, also referred to as misuse detection, operates by matching network traffic or system activities against a predefined database of known attack signatures[2]. This approach is highly effective at identifying known threats but struggles with detecting novel attacks or zero-day exploits, as it relies on the continuous updating of signatures to remain effective. Without pre-existing signatures, signature-based detection cannot recognize new, undocumented threats, and it is less effective at handling multi-connection or more complex attack vectors[3].

In contrast, anomaly-based IDS monitors network activity and flags deviations from established normal behavior as potential threats. This method is more adaptable than signature-based systems and has the ability to detect new and unknown attacks. However, anomaly-based IDS tends to generate a high rate of false positives, which leads to unnecessary alerts and increases the workload for system administrators. The challenge of minimizing false positives has driven the search for more intelligent detection mechanisms, leading to the integration of machine learning into IDS[4].

IDS can also be classified based on functionality into two categories: Network-Based IDS (NIDS) and Host-Based IDS (HIDS). NIDS monitors traffic across an entire network, often using signature-based

methods such as Snort, which detect attacks based on known signatures. HIDS, on the other hand, focuses on monitoring activities on individual hosts within a network, often relying on anomaly-based detection techniques.

In addition to the primary detection approaches, hybrid detection techniques, also known as specification-based detection, combine elements of both signature-based and anomaly-based detection. Hybrid IDS aim to leverage the strengths of both methods while minimizing their weaknesses[5, 6]. These models can be applied to both NIDS and HIDS, providing a more comprehensive defense against a wider range of threats.

Overall, while misuse detection is effective at detecting known attacks with a low rate of false positives, it is limited in its ability to identify new or emerging threats. Anomaly detection, though powerful in detecting unknown attacks, suffers from high false alarm rates and requires significant computational resources to create accurate behaviour profiles. Understanding the strengths and limitations of these methods is crucial for developing IDS frameworks suited to specific network environments, where balancing accuracy, complexity, and response time is essential.

Ongoing research continues to focus on improving the efficiency of IDS, aiming to reduce false alarms while enhancing detection accuracy. The integration of machine learning into IDS offers promising advancements in achieving this balance, as it allows systems to adapt to evolving threats and improve performance across different network environments.

2.2 Classification of Machine Learning

Machine learning can be divided into four major categories, each focusing on distinct tasks and learning each addressing different types of tasks and learning approaches:

- **Supervised Learning**: In supervised learning, labeled datasets are utilized to train models that predict results based on the given input. The aim is to develop a function that links inputs to outputs. This approach is often used for tasks like classification and regression. Algorithms commonly applied in this category include decision trees, support vector machines, and neural networks[7].
- **Unsupervised Learning**: Unsupervised learning operates on data without labels, with the objective of uncovering hidden patterns or structures within the data. Clustering and dimensionality reduction are key techniques under this paradigm, and algorithms like k-means clustering and PCA are commonly employed [8]. Use cases include grouping customers and identifying anomalies.
- **Reinforcement Learning**: Reinforcement learning (RL) deals with decision-making in environments, an agent learns through interaction with its surroundings, receiving rewards or penalties depending on its actions. The objective is to optimize the total rewards over time. Reinforcement learning has been effectively used in fields like robotics, game AI, and autonomous vehicles [9].
- **Semi-Supervised Learning**: Semi-supervised learning combines a small amount of labeled data with a large amount of unlabeled data to improve model accuracy. It is especially useful when labelling data is expensive or time-consuming. Algorithms like self-training and label propagation are commonly used in this approach [10]. Examples include medical image analysis and natural language processing.

3. Methodology for Integrating Machine Learning into Intrusion Detection Systems

This chapter presents the methodology employed to evaluate the performance of IDS enhanced with ML models. The analysis focuses on comparing IDS solutions with ML techniques, with and without

the use of PCA. The purpose of using PCA is to investigate how dimensionality reduction affects model performance, efficiency, and accuracy in detecting network intrusions.

The various ML algorithms evaluated in this study include Random Forest, SVM, ANN, KNN, DT, NB, and LR. These algorithms are applied to the benchmark dataset, KDD Cup 1999, which is widely used for intrusion detection. The performance of these ML-based IDS frameworks is contrasted against traditional IDS methods, such as signature-based and anomaly-based systems, without ML techniques.

The research workflow shown in Figure 1 illustrates the entire process, from data loading and pre-processing to model training and performance evaluation.

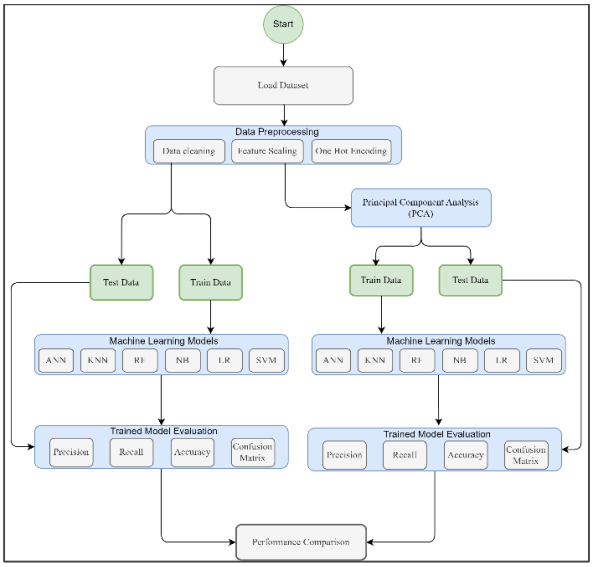

Figure 1: proposed research workflow

3.1 Dataset Selection

The KDD Cup 1999 dataset is a well-known dataset used to test IDS [11]. It contains a large volume of normal and abnormal network traffic and various types of network intrusions. The dataset is extensively used to evaluate the performance of IDS frameworks. It features numerous attributes, including protocol type, service type, source bytes, and destination bytes, which are used to classify network traffic into normal and attack categories. This dataset has become a standard reference point for benchmarking IDS models, helping to measure their effectiveness in detecting and classifying different types of network intrusions.

3.2 Data Loading

The datasets used in this research is KDD Cup 1999, which is used for evaluating the performance of IDS systems. The dataset consists of labeled network traffic data, where each record is classified as either normal or intrusive. Each record contains multiple features representing network traffic characteristics, such as protocol type, service, duration, and more.

Upon loading the dataset, it is divided into two main sets:

- Training Data: Used for training the machine learning models.
- Testing Data: Used to evaluate the models' performance on unseen data.

By dividing the dataset in this way, we ensure that the models' performance is evaluated on data they have not encountered during training, which provides a reliable measure of generalization.

3.3 Data Preprocessing

Before applying ML algorithms, the dataset must undergo several pre-processing steps to ensure it is clean, correctly formatted, and ready for analysis. The following pre-processing steps are carried out:

a. Data Cleaning

Data cleaning involves handling missing, corrupted, or irrelevant data. Missing values are addressed using techniques such as mean/mode imputation, or rows with missing data are removed if the dataset size allows for it [12].

b. Feature Scaling

Feature scaling is applied to ensure that all numerical features are on the same scale. This is particularly important for distance-based models like KNN and SVM[13]. The feature scaling method deployed a median-based scaling method known as Robust Scaler. Robust scaler uses the interquartile range (IQR) and median score (X_{medium})to absorbs the effects of outliers while scaling. Since The interquartile range (Q3 — Q1) has half the data point, the scaler is convenient for removing outliers during the scaling that might affect your results without having to design separate algorithm for that. The equation below describes the scaling process.

$$X_{new} = \frac{X - X_{medium}}{IQR} \quad (1)$$

c. One-Hot Encoding

It is a widely used technique for handling categorical variables, converting them into numerical format to ensure machine learning models can accurately process and interpret non-numerical data. For features such as protocol type or service type, this method transforms each category into a new binary feature, where each categorical value is represented by a 1 or 0. This approach eliminates potential issues of misinterpreting relationships between categorical values, which can arise with simpler encoding methods like label encoding. By preventing any implicit ordering among the categorical values, One-Hot Encoding allows the model to focus on the presence or absence of specific categories, enhancing its ability to classify network traffic data effectively[14].

3.4 Dimensionality Reduction: Principal Component Analysis (PCA)

To address potential issues arising from the high-dimensionality of the data, PCA is applied [15]. It is used to reduce the dimensionality of the input dataset by transforming the features into a set of linearly uncorrelated components. This not only improves computational efficiency but also reduces the risk of overfitting.

- Pre-PCA Model Comparison: Models are first trained and evaluated using the original dataset.
- Post-PCA Model Comparison: PCA is applied, and the models are re-evaluated on the reduced feature set, allowing a direct comparison of model performance before and after dimensionality reduction.

The original length of the feature vector is 122 and computed the best 20 features from each vector there by reducing its size by a whopping 83.6% using the PCA approach.

3.5 Machine Learning-Based IDS

After evaluating traditional IDS approaches, ML-based IDS frameworks are implemented. The following ML models are applied to the datasets:

a. Random Forest (RF)

It is an ensemble learning method, builds multiple decision trees during training, and merges them according to some rule, majority voting or averaging, to improve accuracy and control overfitting. Random Forest is used to classify network traffic based on features from the dataset. We trained the

algorithm with the "bagging" method. The bagging method combines several learning models to increase the overall result

b. Support Vector Machines (SVM)

SVM is a powerful classification algorithm that identifies the optimal hyperplane separating different classes in a high-dimensional space. It is particularly effective in binary classification tasks, such as distinguishing between normal and intrusive network traffic. We used Radial Basis Function Kernel (RBF) as the learning kernel where similarity between two points in the transformed feature space is an exponentially decaying function of the distance between the vectors and the original input space.

c. Artificial Neural Networks (ANN)

ANN is a flexible model that mimics the behavior of the human brain by using interconnected neurons. It is effective at capturing complex, non-linear relationships in the data. For this study, a multi-layer perceptron (MLP) architecture is employed to classify network traffic. The network has a total of five dense layers with "Relu" activation. Each succeeding dense layer is separated by a "Dropout" legalization layer. The dropout layers prevent overfitting during the training. The summary of the ANN architecture is provided in the table below.

Table 1: Artificial Neural Network (ANN) Architecture and Layer Parameters Model "Sequential"

Layer(Type)	Output shape	Param
Dense (Dense)	(None, 64)	7,872
Droput(Droupout)	(None, 64)	0
Dense_1(Dense)	(None, 128)	8,320
Droupout_1(Droupout)	(None, 128)	0
Desne_2(Dense)	(None, 512)	66,048
Droupout_2(Droupout)	(None, 512)	0
Desne_3(Dense)	(None, 128)	65,664
Droupout_3(Droupout)	(None, 128)	0
Desne_4(Dense)	(None, 1)	129

The ANN model consists of 148,033 parameters (578.25 KB), all of which are trainable. There are no non-trainable parameters (0 B), meaning the entire model participates in learning.

d. K-Nearest Neibour (KNN) Algorithm for IDS

The k-nearest neighbors algorithm, often referred to as KNN or k-NN, is a flexible, non-parametric model used for supervised learning. It classifies or predicts outcomes by looking at the proximity of data points. Though it can be applied to both regression and classification tasks, KNN is most commonly used for classification, based on the idea that similar data points are likely to be found near each other:

(2) $$\sum_{i=1}^{k} \sum_{x \epsilon Si} \| \chi - \mu_{\iota} \|^2$$

This clustering process helps identify unusual patterns in network traffic, making it suitable for detecting unknown or novel attacks in IDS[16]. In the context of machine learning, K-NN aids in pre-processing and feature extraction, grouping similar data points for further analysis or classification.

e. Naïve Bayes (NV) Algorithm for IDS and Machine Learning

Naïve Bayes, a probabilistic classifier based on Bayes' theorem, is another algorithm well-suited for IDS and ML tasks. It calculates the probability of a data point belonging to a certain class, assuming independence between features:

(3)
$$P(C \mid F_1, F_2, \ldots, F_n) \propto P(C) \prod_{i=1}^{n} P(F_i \mid C)$$

Despite its simplicity, Naïve Bayes performs effectively in anomaly detection, identifying attacks by analyzing packet behavior and detecting deviations from normal traffic. Its ability to handle high-dimensional data and reduce false alarm rates makes it a robust choice for IDS[5].

3.6 Model Training

Both the original and PCA-transformed datasets are split into training data and testing data. The training data is used to train the ML models, while the test data is used to evaluate their performance. A standard split of 80% training and 20% testing is used.

3.7 Confusion Matrix

The Confusion Matrix is a valuable tool for understanding the performance of an IDS by offering a clear visual representation of classification outcomes for both normal and attack instances. This matrix categorizes the predicted and actual results of the system, displaying the four possible outcomes as shown in Table 2.

The confusion matrix helps in evaluating an IDS's performance, particularly in imbalanced datasets, where the number of normal traffic instances often outweighs the number of attack instances. This tabular representation assists in assessing key performance metrics such as accuracy, detection rate, and false alarm rate, which are crucial for determining the system's effectiveness. The confusion matrix for a binary classification system, such as an Intrusion Detection System (IDS), is presented in Table 2.

Table 2: Confusion Matrix for IDS Performance

Actual/Predicted	Predicted: Attack	Predicted: Normal
Actual: Attack	True Positive (TP)	Negative (FN)
Actual: Normal	False Positive (FP)	True Negative (TN)

Both traditional IDS methods and ML-based IDS models are evaluated on these metrics to compare their effectiveness in intrusion detection. Special attention is given to reducing false positives, which are a critical issue in real-world IDS implementations.

3.8 Performance Comparison and Analysis

To determine the impact of incorporating ML algorithms into IDS frameworks, the following comparisons are made:

- Traditional IDS vs. ML-based IDS: The performance of signature-based and anomaly-based IDS methods is compared against ML-based systems using Random Forest, SVM, and ANN.
- PCA vs. Non-PCA Models: The effect of dimensionality reduction on model performance is evaluated by comparing models trained on the original dataset with those trained on the PCA-transformed dataset.
- Detection Accuracy and False Positives: The study places particular emphasis on improving detection accuracy while minimizing false positive rates.

This methodology outlines the process of evaluating traditional IDS systems alongside ML-enhanced IDS models. By comparing the performance of signature-based, anomaly-based, and ML-based IDS frameworks on benchmark datasets, the study aims to highlight the advantages of integrating ML into IDS solutions. The experimental results provide insights into which ML models and methodologies are most effective at improving detection accuracy and reducing false positives in modern network environments.

4. Analysis And Results

The purpose of this chapter is to evaluate the performance of the selected machine learning models with and without the application of Principal Component Analysis (PCA). The primary objective is to demonstrate how dimensionality reduction through PCA improves the overall performance, particularly in terms of accuracy, for Intrusion Detection Systems (IDS).

4.1 Model Performance Without PCA

The performance of various machine learning models, including ANN, KNN, DT, NB, LR, and SVM, was first evaluated without the application of PCA. Table 3 shows the accuracy, precision, recall, and confusion matrix for each model. For ANN, the model was trained for 50 epochs, and the training and validation accuracy are depicted in Figure 2.

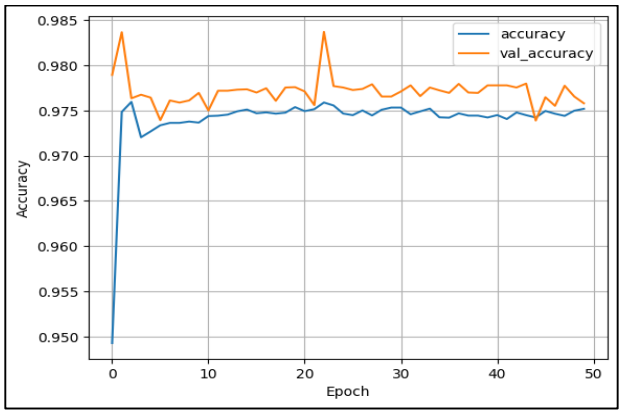

Figure 2: Training and validation accuracy for ANN

Table 3: Model Performance without PCA

Model	Accuracy	Precision	Recall	Confusion Matrix (FP, FN)
ANN	97.56%	98.66	95.80	(143,26)
KNN	99.05	99.22	98.73	(157, 111)
RF	99.99	99.99	99.99	(21,5)
NB	91.80	92.63	89.48	(1264, 851)
LR	88.19	83.92	92.25	(938,2116)
SVM	97.20	97.89	96.04	(471, 274)

From the results, it can be observed that while the models perform reasonably well, they may suffer from overfitting or inefficiencies due to the high dimensionality of the feature space. Random Forest (RF) proves to be the most reliable model without PCA, showing near-perfect accuracy, precision, and recall, making it ideal for tasks requiring both. KNN also performs well, though with slightly lower recall, making it a strong alternative for accuracy-focused tasks. ANN and SVM provide comparable accuracy but have lower recall than RF and KNN, which might suit tasks prioritizing precision. In

contrast, NB and LR show weaker performance due to high false positives and negatives, making them less effective for this task.

4.2 Applying PCA to Reduce Dimensionality

The results presented in Table 4 highlight the varying impacts of Principal Component Analysis (PCA) on the performance of different machine learning models. Random Forest with PCA delivered near-perfect results, achieving 99.99% accuracy, precision, and recall, indicating its strong resilience to dimensionality reduction. K-Nearest Neighbors with PCA also performed well, with 98.99% accuracy, although a slight decrease in recall (98.62) led to an increase in false negatives. Artificial Neural Networks and Support Vector Machine, both with PCA, maintained solid performance, with only minor declines in recall (95.44 and 95.27, respectively), confirming their adaptability to PCA. Logistic Regression with PCA showed notable improvement, with an accuracy of 89.56% and a recall of 92.04%, indicating that PCA enhanced its ability to detect positive instances.

In contrast, Naive Bayes with PCA struggled significantly, with recall dropping to 55.90%, resulting in a substantial increase in false negatives. This shows that Naive Bayes is less suited for classification tasks when PCA is applied. Overall, Random Forest with PCA demonstrated the most reliable performance, followed by K-Nearest Neighbors, Support Vector Machine, and Logistic Regression. However, Naive Bayes showed a marked degradation in performance after PCA, highlighting its limitations with dimensionality reduction.

Table 4: Model Performance with PCA

Model	Accuracy	Precision	Recall	Confusion Matrix
ANN+PCA	96.08%	97.68	95.44	(148,29)
KNN+PCA	98.99	99.21	98.62	(160, 108)
RF+PCA	99.99	99.99	99.99	(34,10)
NB+PCA	77.03	91.27	55.90	(5183, 666)
LR+PCA	89.56	86.39	92.04	(973, 1754)
SVM+PCA	96.63	97.41	95.27	(557, 319)

4.3 Impact of PCA on Model Performance

The graphs in Figure 3 demonstrate the impact of PCA on accuracy, precision, and recall across the different models. From the graph, we can observe that RF consistently achieves the highest values across all three metrics, both with and without PCA, confirming it as the most reliable model for this IDS task.

- **K-Nearest Neighbors (KNN)** also performs well after PCA, with high precision and recall values. However, there is a slight increase in false negatives, Figure 4, suggesting a minor reduction in recall post-PCA. This is consistent with the performance comparison in Figure 5, where KNN maintains high overall accuracy but slightly lower recall.
- **Artificial Neural Networks (ANN)** and **Support Vector Machine (SVM)** exhibit similar performance patterns, with relatively strong precision and recall values after PCA, as illustrated in Figure 3. The slight reduction in recall for ANN Figure 3 is accompanied by a corresponding increase in false negatives (Figure 4). However, both models continue to perform reliably even after dimensionality reduction.

4.4 Models with Limited Improvement from PCA

Naive Bayes (NB) shows a sharp drop in recall and an increase in false negatives after PCA, as seen in Figure 4. This suggests that NB struggles to detect positive instances in reduced dimensions, despite

retaining reasonable precision. Logistic Regression (LR) sees a minor recall improvement post-PCA but still produces a considerable number of false positives and negatives, as shown in Figures 3 and 4. While PCA brings slight balance, LR remains less effective compared to Random Forest and SVM.

4.5 Analysis of False Positives and False Negatives

Figure 5-4 shows the false positives (FP) and false negatives (FN) for each model. Random Forest, K-Nearest Neighbors, and Support Vector Machine maintain low FP and FN rates, demonstrating strong performance even after PCA, which is vital for effective detection in IDS systems. On the other hand, Naive Bayes and Logistic Regression struggle, with Naive Bayes showing a notable rise in false negatives, reflecting its poor recall after PCA, as also seen in Figures 3 and 4.

4.6 Performance with and without PCA

The performance comparison in Figure 5 further highlights how PCA affects model performance. Random Forest and SVM retain high accuracy, precision, and recall in both PCA and non-PCA scenarios. K-NN and Artificial Neural Networks show slight trade-offs in recall after PCA but maintain competitive accuracy and precision. These models effectively balance the trade-off between dimensionality reduction and classification accuracy.

Conversely, Naive Bayes and Logistic Regression show less favourable outcomes, particularly with increased false negatives. Naive Bayes, in particular, experiences a significant drop in performance after PCA, indicating that it is not well-suited for dimensionality reduction in this context.

4.7 Discussion and comparison

The results clearly indicate that PCA plays a significant role in reducing the computational cost of IDS models and produces excellent competitive performance compared to the non-PCA approach. Reducing dimensionality allows models to focus on the most critical aspects of the data, improving both detection rates and computational efficiency. While PCA does lead to any majore information loss, this trade-off is acceptable given the overall improvement in model accuracy and reduction in false positives.

However, it is also important to note that PCA may not always yield better results for all machine learning models. For instance, models like Naive Bayes did not show a dramatic improvement, suggesting that further optimizations may be necessary.

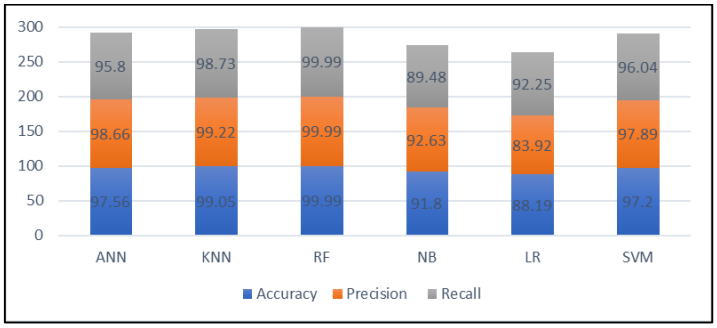

Figure 3: performance comparison between different ML algorithms

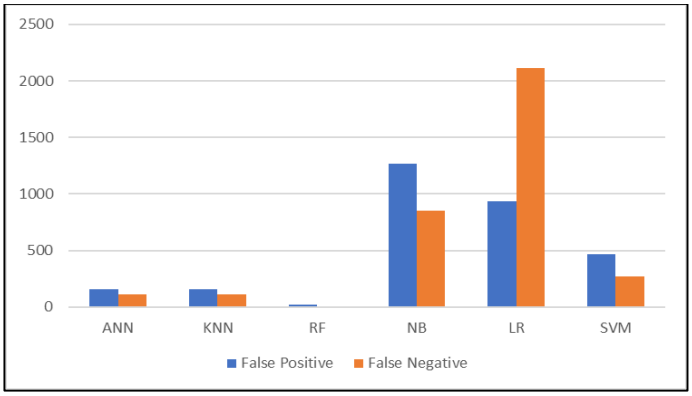

Figure 4: FP and FN comparison between different ML algorithms

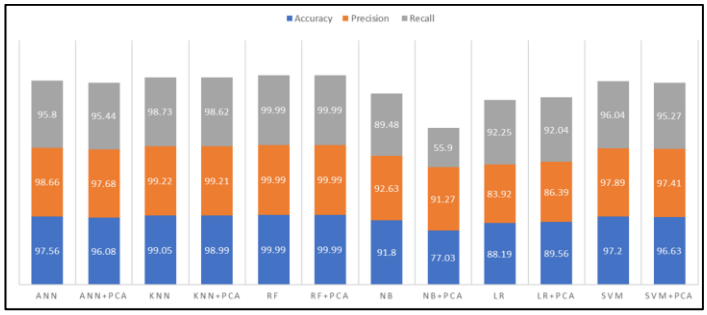

Figure 5: performance comparison between different ML algorithms with and without PCA

5. Conclusion

In this study, we examined the impact of using PCA to reduce the dimensionality of the KDD Cup 1999 dataset. By reducing the original 122 features by 83.6%, down to just 20, we aimed to evaluate the effect of this reduction on the performance of several machine learning models.

Models such as RF and KNN stood out by maintaining near-perfect accuracy and recall even after the substantial feature reduction. These findings suggest that these models can effectively handle high-dimensional datasets, even when reduced, without sacrificing performance. On the other hand, SVM and ANN showed minor performance drops but still performed well, indicating they are quite resilient to feature reduction, though they benefit slightly from having more features. LR, interestingly, saw a slight improvement with PCA, likely due to a reduction in noise or redundancy. In contrast, NB was significantly impacted by PCA, with a sharp decline in recall, highlighting that it relies more on the complete feature set for accuracy. Overall, PCA proved to be an effective way to reduce dimensionality without overly compromising the performance of most models, but its effect varies across algorithms. For models like Naive Bayes, maintaining a larger feature set seems critical for optimal performance.

6. Future Work

For future research, it would be useful to test PCA with different levels of feature reduction, such as retaining 30%, 40% or more of the features, to see how this impacts performance across different models. This could offer a clearer understanding of how sensitive each model is to the degree of dimensionality reduction. Additionally, experimenting with other feature selection techniques or hybrid approaches that combine PCA with other methods might help further improve model performance, particularly for models like Naive Bayes that were negatively affected by extreme reduction.

7. Author's Contribution:

We confirm that all named authors have read and approved the manuscript. We also confirm that each author has the same contribution to the paper. We further confirm that all authors have approved the order of authors listed in the manuscript.

8. Conflict of Interest:

There is no conflict of interest for this paper.

References

[1] D. E. Denning, "An Intrusion-Detection Model," *IEEE Transactions on Software Engineering*, vol. SE-13, no. 2, pp. 222-232, 1987, https://doi.org/10.1109/TSE.1987.232894

[2] Snort. "The Open Source Network Intrusion Detection System." https://www.snort.org (accessed 27/0, 2024).

[3] R. Bace and P. Mell, "NIST special publication on intrusion detection systems," *National Institute of Standards and Technology*, vol. 16, 2001.

[4] V. Jyothsna, R. Prasad, and K. M. Prasad, "A review of anomaly based intrusion detection systems," *International Journal of Computer Applications*, vol. 28, no. 7, pp. 26-35, 2011.

[5] S. M. Hussein, "Performance Evaluation of Intrusion Detection System Using Anomaly and Signature Based Algorithms to Reduction False Alarm Rate and Detect Unknown Attacks," in *2016 International Conference on Computational Science and Computational Intelligence (CSCI)*, 15-17 Dec. 2016 2016, pp. 1064-1069, https://doi.org/10.1109/CSCI.2016.0203.

[6] S. M. Hussein, F. H. M. Ali, and Z. Kasiran, "Evaluation effectiveness of hybrid IDS using Snort with Naïve Bayes to detect attacks," in *2012 Second International Conference on Digital Information and Communication Technology and it's Applications (DICTAP)*, 16-18 May 2012 2012, pp. 256-260, https://doi.org/10.1109/DICTAP.2012.6215386.

[7] I. Goodfellow, "Deep learning," ed: MIT press, 2016.

[8] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[9] A. Agarwal, N. Jiang, S. M. Kakade, and W. Sun, "Reinforcement learning: Theory and algorithms," *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, vol. 32, p. 96, 2019.

[10] X. Zhu and A. B. Goldberg, *Introduction to semi-supervised learning*. Springer Nature, 2022.

[11] K. C. 1999. "KDD Cup 1999 Dataset." https://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (accessed September 28, 2024).

[12] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. " O'Reilly Media, Inc.", 2022.

[13] C. Cortes, "Support-Vector Networks," *Machine Learning*, 1995.

[14] F. Farahnakian and J. Heikkonen, "A deep auto-encoder based approach for intrusion detection system," in *2018 20th International Conference on Advanced Communication Technology (ICACT)*, 2018: IEEE, pp. 178-183.

[15] I. T. Jolliffe and J. Cadima, "Principal component analysis: a review and recent developments," *Philosophical transactions of the royal society A: Mathematical, Physical and Engineering Sciences*, vol. 374, no. 2065, p. 20150202, 2016.