Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

A Machine Learning-Driven Cloud Intrusion Detection: Impact of Feature Engineering and Dimensionality Reduction on Classification and Optimization

Kenda Alamer[1], Abdulaziz Aldribi[2]
[1,2]*Department of Computer Science, College of Computer, Qassim University, Buraydah, Saudi Arabia*
411200226@qu.edu.sa, aaldribi@qu.edu.sa

ARTICLE INFO

Received: 29 Dec 2024

Revised: 15 Feb 2025

Accepted: 24 Feb 2025

ABSTRACT

Cloud computing has emerged as one of the most rapidly advancing domains in modern information technology, enabling scalable and efficient resource utilization across diverse industries. However, with this expansion comes serious cyberattacks which can disrupt critical sectors, such as healthcare, telecommunications, and finance, making significant challenges to intrusion detection systems. To address these challenges, this paper presents a robust intrusion detection framework designed to enhance security in cloud computing environments. The contributions of this work begins with the selection of a benchmark dataset representative of real cloud-based network traffic, followed by extraction of new critical features to identify the most relevant attributes for model development. We then leverage Principal Component Analysis (PCA) for feature reduction to enhance model efficiency. To detect intrusions, we employ multiple Non-parametric and Parametric Learning Models and systematically evaluate their performance under four configurations: feature extraction with and without PCA, and PCA with and without feature extraction. Furthermore, to improve classification performance, we conduct hyperparameter optimization on the best performing algorithm. Finally, we conduct a comprehensive comparison with state-of-the-art intrusion detection techniques, demonstrating the efficacy of our approach. Experimental results demonstrate that our optimized Multilayer Perceptron (MLP) model achieves the highest detection accuracy. This work not only advances intrusion detection in cloud networks but also provides insights into feature engineering and algorithmic optimization for improved cybersecurity.

Keywords: Cloud Computing, Machine Learning, Network Security, Feature Extraction , Optimization

INTRODUCTION

The growth of network devices connected with Internet has led to a large number of applications that rely on processing and sharing of data. Thus, cloud computing has become of utmost importance in this era of big data and Internet of Things. Cloud computing has many facets that can enable fast processing and storage of data. This includes parallel and distributed computing, fog computing and edge computing.

Cloud computing offers several advantages to industries and businesses. Industries need not to worry about data storage and processing of the data as cloud computing performs these services on their behalf. Cloud computing in conjunction with virtualization provides a way to manage computing resources efficiently, thus resulting in resource maximization.

While cloud computing has several benefits, it also faces few challenges for successful implementation and to provide efficient services. The most important concern for cloud computing is security of data in the cloud. In the current age of information technology, the number and type of attacks have grown significantly. A number of attacks including data integrity, eavesdropping, network jamming, and denial of service can be launched to compromise the cloud systems and limit their capabilities Aldribi et al. (2018). While conventional security concerns remain relevant,

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Catteddu and Hogben (2009) have highlighted specific threats associated with cloud computing. These including the loss of governance, insufficient data protection, compliance vulnerabilities, insider threats, compromised management interfaces, and incomplete or insecure data removal. An additional critical issue, once data is transferred to cloud infrastructure, users face the risk of losing control over their information. This reliance on cloud resources also exposes data integrity and confidentiality to potential threats Aldribi et al. (2020a). Modi and Patel (2013) categorize cloud attacks into two types based on their source: insider and outsider attacks. Insider attacks are carried out by individuals which are trustable,

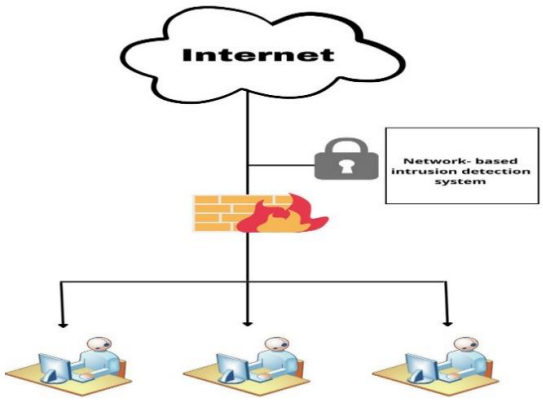

Figure 1. Network Attack Detection System

such as nodes that are located at the root in the hierarchy of the cloud. Conversely, outsider attacks aim to disrupt the computing mechanism by exploiting vulnerabilities within the cloud system.

To protect data in the cloud computing network, there's a growing need for monitoring systems that track events and detect intrusions. Thus, there is a need to develop Intrusion Detection System (IDS) which can be hardware or software framework. Such IDS helps in identifying different threats and protecting against network-based attacks Mehmood et al. (2013). Figure 1 illustrates a typical intrusion detection system, constructed using various techniques. The development of IDS faces several challenges, leading to the use of various approaches like rule-based, statistical, and fuzzy logic methodologies to create effective intrusion detection systems over time.

A network Intrusion Detection System is designed to differentiate between normal and unusual network behavior, effectively identifying intrusions using Machine Learning (ML) techniques. Recent studies have shown that ML can achieve high accuracy in detecting and managing new forms of intrusion Firdausi et al. (2010).

Since cloud computing has gained significant popularity in the IT industry and has attracted interest from both government and academic sectors. Consequently, there's a pressing need for robust security mechanisms to address the unique threats faced by cloud environments.

However, many existing systems rely on datasets designed for traditional network architectures, which don't accurately reflect real world cloud settings.

The proposed direction for further research in industry and academia is to apply Machine Learning algorithms to datasets derived from actual cloud environments. This approach aims to create new intrusion models to detect and classify security breaches within cloud computing settings. By focusing on real cloud data, these new models can offer more accurate and effective ways to protect cloud infrastructure.

The goal of this paper is to develop a classification mechanism for detecting intrusions in cloud computing by leveraging efficient machine learning algorithms to achieve optimal results across various models. The key objectives are as follows:

- Extract new features from cloud network traffic and utilize Principal Component Analysis (PCA) for feature reduction.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- Employ selected ML algorithms to classify intrusions within the cloud computing network.
- Apply different techniques to assess their impact on ML performance Such as: feature extraction without PCA, feature extraction with PCA, no feature extraction but with PCA, and no feature extraction without PCA.
- Optimize the best-performing algorithm to achieve high classification accuracy.
- Enhance algorithm accuracy by focusing on important extracted features.
- Compare the proposed approach with current state-of-the-art techniques.

The rest of this paper is structured as follows: Section 2 presents a literature review, discussing feature engineering and existing intrusion detection system approaches. Section 3 describes the dataset used in this paper, along with an introduction to the machine learning algorithms employed. Section 4 outlines key performance metrics and presents the paper results. Section 5 provides the conclusions and future opportunities.

LITERATURE REVIEW

The cloud computing security is essential for its implementation and providing service to different applications. Cloud systems need to maintain authenticity and confidentiality to provide secure data storage and computing. To achieve this, techniques based on analysis of system logs, configurations, and network traffic are developed to enhance cloud computing security. To develop an efficient Intrusion Detection System for a particular scenario and network requires a dataset taken from the considered network to make sure that the IDS is accurate. The dataset allows researchers to test and make changes in the design of IDS, thus improving their reliability for threat detection.

Intrusion Detection Models

For intrusion detection, two major models are used. The first is the anomaly detection and second is the misuse detection. Misuse intrusion detection is also called as signature-based or knowledge-based detection. It finds intrusions by comparing observed patterns with attack signatures that are known. This model has good performance in accurately recognizing many types of attacks but fails in the case of Zero-day attacks. Anomaly detection works by developing a normal behavior model and identifies abnormalities as potential intrusions. As anomaly detection has a behavior-based approach, it can identify new attacks but can cause higher number of false positives because not all abnormalities are intrusions.

The work by Modi et al. (2012) used two types of intrusion detection techniques. The first is Snort which is misuse-based technique and is designed to detect known intrusions. The Anomaly detection is the second technique that uses any changes from expected behavior and mark them as intrusions.

The authors Mighan and Kahani (2018) proposed a hybrid approach to intrusion detection. The proposed work used a Stacked Auto Encoder (SAE) for extracting features. In addition, a Support Vector Machine (SVM) is used for classification. This combination outperformed SVM alone, with accuracy results for SAE with SVM at 90.2%, compared to PCA with SVM at 85.6

Pandeeswari and Kumar (2016) developed a detection mechanism for new attacks and identify its patterns. The idea relies on Fuzzy C-means Clustering-Artificial Neural Network (FCMANN). The developed technique was implemented on simulator for clouding computing (Cloudsim). The performance was tested based on Artificial Neural Networks (ANN) and Naïve Bayes (NB) to test the performance in presence of different attacks.

The work in Denning (1987) developed a system based on rules with statistical anomaly detection elements. Verwoerd and Hunt (2002) also explored statistical models for intrusion detection. Aldribi et al. (2018) discussed two other types of IDS models, the first one is network-based whereas the second one is the host-based. To assist IDS in identifying abnormal network traffic, Alshammari and Aldribi (2021) proposed a detection architecture using a Machine Learning model trained on various attacks to distinguish between anomalies and regular traffic.

Finally, the work in Ikram and Cherukuri (2016) proposed a model for intrusion detection that combines SVM and PCA. The model also optimizes and tuned the SVM parameters, aiming for optimal performance

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

IDS Datasets

One of the limitations of the NSL-KDD Dhanabal and Shantharajah (2015) and DARPA Subba et al. (2016) datasets is that they are based on conventional network architectures, which may not accurately represent the changing landscape of computing, especially with the rise of cloud computing. Developing effective IDS for cloud environments is challenging because there is a shortage of publicly available datasets tailored to cloud intrusion detection.

The work in Aldribi et al. (2018 2020a) focused on this gap by putting online a dataset for intrusion detection in cloud, known as the Information Security and Object Technology Cloud Intrusion Dataset (ISOT-CID). This dataset is derived from various cloud environments and sources, offering multiple formats. Proper analysis and maintenance of datasets are crucial for research, and large datasets often require preprocessing to create structured data suitable for different algorithms.

Modi et al. (2012) used Eucalyptus infrastructure to simulate a cloud computing environment and utilized the KDD CUP IDS dataset to evaluate their work, achieving a detection rate of 96.06 Khan et al. (2019) applied the DARPA 1998 dataset in their research, achieving an accuracy rate of 99.2

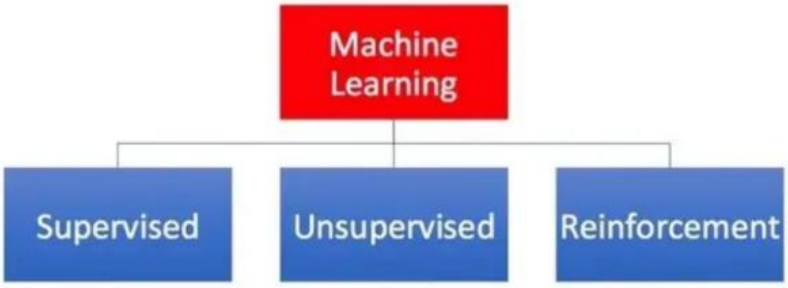

Figure 2. Classification of Machine Learning

Pandeeswari and Kumar (2016) also used the KDD CUP dataset for algorithm evaluation. Their dataset processing included three steps: (1) preprocessing to make the data suitable for feature extraction, (2) applying Stacked Auto-Encoder to extract latent features from the 42 original features, and (3) applying Support Vector Machine to classify records as attack or benign.

Alshammari and Aldribi (2021) utilized a dataset that contained both malicious and benign traffic to evaluate their detection architecture.

Chou and Wang (2016). evaluated an adaptive network-based intrusion detection system for the cloud using the DARPA 2000 and KDD Cup 1999 datasets. In the DARPA dataset, they achieved a 95% detection rate with a 4.5% false positive rate. In the KDD Cup 1999 dataset, they obtained a 90% detection rate with a 5% false positive rate.

Jamadar (2018) conducted dataset featuring to reduce the number of features. The idea was based on Recursive Feature Elimination (RFE). The study used the dataset in CIDS 2017, achieving high accuracy levels reaching 99.9.

Sakr et al. (2019) evaluated their proposed system using the NSL- KDD dataset. Their results indicated that incorporating feature selection with SVM led to an approximately 6% increase in accuracy compared to SVM without Binary Based Particle Swarm Optimization (BPSO).

These different datasets and approaches highlight the challenges and ongoing efforts in creating effective IDS for cloud computing environments, emphasizing the need for more cloud-specific datasets and improved methods for feature extraction and classification.

ML Techniques

Machine Learning (ML) can be categorized into three main types: supervised learning, unsupervised learning, and reinforcement learning. In supervised learning, each record in a dataset is associated with a class label that indicates the category to which it belongs. In unsupervised learning, the data lacks specific labels, requiring algorithms to

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

discover patterns or clusters. Reinforcement learning combines elements of both supervised and unsupervised learning, where a small portion of the dataset has labels, while the majority does not.

Different supervised ML algorithms, as illustrated in Figure 2, are often used to construct IDS to classify samples in a dataset based on attack type. The motivation behind this research lies in the need for effective network attack classification systems in cloud computing, given the rapid growth of cloud services and the demonstrated effectiveness of machine learning in addressing system security challenges.

Therefore, this paper aims to harness the capabilities of ML to develop an Intrusion Classification Framework for Cloud Computing, seeking to improve the accuracy and efficiency of identifying security threats in cloud environments.

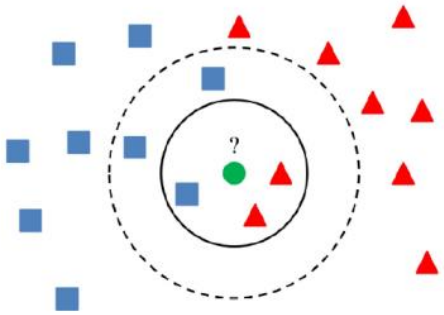

Figure 3. K-Nearest Neighbors Structure

Artificial neural networks (ANNs) are widely recognized today, but their roots trace back to 1943, when a computational model of biological neural networks was introduced by Mcculloch and Pitts (1990). The current popularity of ANNs is a result of years of evolution and enhancement. The core concept of neural networks is inspired by the way the human brain processes complex tasks using prior experiences. ANNs consist of interconnected nodes, or neurons, with links between them responsible for transmitting information from the input layer to the output layer. The learning process involves feeding data from the input to the output layer through weighted connections, with the weights adjusted based on the error produced by the network. ANNs have various structures depending on the types of data and learning applications. Multilayer Perceptron (MLP) is a well-known ANN model used in supervised learning, designed to train datasets with labeled data.

The Multilayer Perceptron is a type of feedforward artificial neural network that includes at least one hidden layer. The computational process follows a unidirectional path, progressing sequentially from input nodes through intermediate hidden layers before reaching the output layer. Each layer consists of multiple neurons, with each neuron in one layer being fully connected to all neurons in the subsequent layer. For binary classification scenarios, the output layer is implemented with a singular neuron, whereas multi-class classification requires multiple output neurons. The feedforward mechanism resembles the perceptron model, where neuron activations in hidden layers are computed based on weighted combinations of inputs from the preceding layer Rosenblatt (1958).

The Support Vector Machine (SVM) is a Machine Learning algorithm renowned for its effectiveness in binary classification problems. Recently, SVM has been employed to tackle various challenges, including learning, prediction, and classification Mulay et al. (2010). The central objective of SVM is to develop a computationally efficient mechanism for constructing optimal hyperplanes in high-dimensional feature spaces. These hyperplanes, known as decision boundaries, are used to separate datasets into different groups by maximizing the margin between the decision boundary and the data points in each class. Increasing this margin helps to improve the accuracy of classification for new examples.

In SVM, the dimensionality of the hyperplane varies depending on the number of features. For example, if there are three features, the hyperplane is two-dimensional. As the number of features grows, so does the hyperplane's dimensionality. The support vectors, which are specific data points, play a key role in defining the position and orientation of the decision boundary.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

K-Nearest Neighbors (KNN) is a supervised, instance-based classification algorithm used for both classification and regression problems. It classifies instances in the feature space by identifying the nearest training samples to a given input instance. KNN is non-parametric, meaning it doesn't make any assumptions about the data's distribution or underlying structure. It's also considered a lazy algorithm because it doesn't have a training phase; instead, it makes classifications by computing distances between instances at the time of prediction. In KNN, the classification is determined by a fixed number of nearest training samples, typically referred to as k. When a new sample is introduced, the algorithm calculates its distance from the k nearest training samples and assigns the sample to the class with the majority representation among those k nearest samples. The choice of k is crucial: a small value for k can lead to lower accuracy due to sensitivity to noise, while a large value might result in overfitting. Therefore, the selection of k should be carefully considered.

KNN uses various distance metrics to calculate the proximity between instances, such as Hamming distance, Manhattan distance, Minkowski distance, and Euclidean distance Chumachenko (2017). One example of KNN based classification is shown in Figure 3. The test sample (green dot) is classified as either a red triangle or blue square. If k=3 (indicated by the inner solid-line circle), it is assigned to the red triangles because the inner circle contains two red triangles and one blue square. If k=5 (represented by the outer dotted-line circle), it would be classified as a blue square, with three blue squares versus two red triangles within the outer circle.

The optimal value for k varies depending on the data. Generally, higher values of k reduce the impact of noise on classification but can blur class boundaries. Different heuristic methods can help determine the ideal value for k. The simplest case is when k=1, known as the nearest-neighbor approach, where the classification is determined by the closest training sample's class.

Modi et al. (2012) utilized a Bayesian classification algorithm to detect network anomalies. The Bayes classifier makes decisions about whether a given network event is an intrusion based on observations from previous occurrences.

Kannan et al. (2012) conducted a study that combined Fuzzy Support Vector Machines (FSVM) with a genetic-based feature selection method. This approach resulted in a low false alarm rate and a high detection rate for identifying anomalies. Their results indicated that detection accuracy improves when fuzzy SVM is applied to fewer than 17 features, likely due to reduced conflicts among fuzzy rules in intrusion detection applications.

Pandeeswari and Kumar (2016) found that the Fuzzy C-means Clustering-Artificial Neural Network (FCMANN) algorithm yields the best results due to its high performance across different attack types. Specifically, the average performance in detecting probing, DoS, U2R, and R2L attacks is a detection rate of 98.6% with a false positive rate of 3.06%. Intrusion detection methods vary significantly, including statistical models like those proposed by Verwoerd and Hunt (2002) and ML-based approaches like those used by Nolan and Lally (2018).

Chou and Wang (2016) employed an unsupervised learning algorithm to create a decision tree-based detection model aimed at identifying anomalies in unlabeled data. This approach can be useful for exploring unusual patterns without prior labeling, aiding in the detection of emerging threats and new types of attacks.

Dimensionality Reduction

Principal Component Analysis (PCA) is a widely used technique for reducing the dimensionality of features, applicable across various research fields and features. This can be solely used to reduce number of dimensions or combined with other algorithms for better performance. The work in Salo et al. (2019) proposed an approach to enhance feature selection by PCA and executing Information Gain (IG) to reduce the number of features in the NSL-KDD dataset, followed by applying SVM as an ensemble classifier. Their results demonstrate that using dimension reduction methods combined with ensemble techniques significantly outperforms individual methods.

Another approach, presented in Ravale et al. (2015), utilizes K-means clustering to group the features, then employs SVM as a classifier to groups the KDDCUP'99 dataset as follows: R2L, U2R, DOS, Normal, and Probe. Moreover, further reduction in features on NSL-KDD dataset is carried out by Naïve Bayes, Correlation-based Feature Selection, SVM and J48 algorithms. Performance evaluation shows that highest accuracy is achieved by J48 algorithm.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Overall, these studies highlight the importance of feature selection and dimensionality reduction in improving the accuracy and efficiency of machine learning-based intrusion detection systems. PCA and other reduction techniques can be instrumental in enhancing the performance of various classifiers.

METHODOLOGY

The goal of this paper is to develop a cloud computing intrusion detection framework using three efficient machine learning techniques. For the algorithms, we employ the Multilayer perceptron, K-Nearest Neighbors, and Support Vector Machine learning techniques as our proposed classifiers. KNN learning technique has the ability to classify data when the boundary lines between them is non-linear. KNN can also adapt decision boundary lines by varying the K value. In other words, it constantly evolves with new data. On the other hand, SVM works better for data that has multiple dimensions and also has high distance of separation between classes. Lastly, MLP is more suited for label or class based classification problems.

In this paper, the ISOT-CID dataset is employed, with new features being extracted and subsequently provided as input to all the algorithms. Moreover, the best algorithm is selected based on classification accuracy rate.

Much of the prior research on cloud intrusion detection systems focuses on conventional datasets like DARPA 1998 or KDD99 datasets. Moreover, some cloud network traffic research datasets are not publicly accessible at times due to privacy concerns. These issues prevent cloud researchers from accessing a comprehensive real-world cloud intrusion dataset. In 2018, Aldribi et al. (2018) mitigated these dataset challenges by introducing the ISOT-CID cloud intrusion detection dataset, which represents an authentic cloud-specific dataset and is openly available for researchers to use. The dataset comprises a diversity of data sources, including memory, CPU performance data, system logs, and network traffic. In this paper, we will describe the ISOT-CID dataset, introduce briefs and details about the collection procedures and environment used in order to obtain this dataset.

ISOT-CID DATASET OVERVIEW

Compute Canada which is a cloud service provider and non-profitable organization provides computational requirements and assistance for researchers. It introduced two cloud environments: the east cloud, hosted by Sherbrooke University, and the west cloud, hosted by Victoria University. The proposed data is collected from an open-source platform called OpenStack which enables cloud providers to control computing process and storage, a network resource.

The collection of the ISOT-CID dataset involves two phases: Phase 1 and Phase 2. Data essential for the dataset is gathered from various cloud layers, including VM, Network, and Hypervisor layers. There are three hypervisor nodes in the ISOT-CID project environment: node A, node B, and node C. In the ISOT-CID collection environment, the researchers design and develop collector

Table I Some Raw Features Extracted from the Hypervisor Network Traffic

Date time of first packet	TimeFirst
Date time of last packet	TimeLast
Flow duration	Duration
Number of transmitted packets	NumPktsSnt
Number of received packets	NumPktsRcvd
Number of transmitted bytes	NumBytesSnt
Number of received bytes	NumBytesRcvd
Minimum layer 3 packet size	MinPktSz
Maximum layer 3 packet size	MaxPktSz
Average layer 3 packet size	AvePktSize
Standard deviation layer 3 packet size	StdPktSize
Minimum IAT	MinIAT
Maximum IAT	MaxIAT
Average IAT	AveIAT
Standard deviation IAT	StdIAT
Send packets per second	Pktps
Byte stream asymmetry	Bytps

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

agents in order to collect heavy and specialized data, such as memory and TCP dumps into three cloud layers as follows: VM or Instance-based agents,

Hypervisor-based agents, and Network-based agents. Therefore, each of the VMs has an instance-based agent that records VM's runtime data and sends it to the ISOT-CID lab log server. in addition, each hypervisor has a hypervisor-based agent and tries to record the hypervisor's runtime data, then sends this data to the log server. Finally, the network- based agents in addition, gather runtime data on hypervisor network traffic which is passed to the log server. By distributing the overall cloud stack tiers of the agent, they are able to create a diverse and comprehensive cloud dataset from various resources, as mentioned in Aldribi et al. (2018). Table I shows some features we extracted from Tranalyzer, such as Date time of first packet, Inter-Arrival Time (IAT) Minimum.

To classify data, we follow a list of table actions carried out during documenting ISOT-CID cloud intrusion detection benchmark dataset attack scenarios. This list can be found in Aldribi et al. (2020b) where you can find the attack or the action type file, source and destination IP addresses, and time of the event available in the dataset. According to this list, all kinds of normal and malicious IP addresses for the source and the destination are determined.

The dataset of ISOT-CID has numerous labels, including benign entries, various types of attacks, and different kinds of intrusion. Thereby, we combined numerous attacks under a common attack category. We deleted the rows that related to the attack which contain a few samples.. In the ISOT-CID dataset, we faced the issue of having some missing data which cannot be used. To address this challenge, we identified features that do not provide meaningful values and remove them, as they would only have minor contribution to machine learning tasks. Moreover, to determine these meaningless features, we rely on the following principle: if the number of examples containing zero value is greater than half of the number of samples, they are removed from the dataset. We in addition, relied on deleting the features by looking at the correlation between features and the class label to find the highly correlated features thus we can delete them and keep one of them only. We do that by plotting the correlation coefficient for all combinations of features and finally deleting the highly correlated features from the dataset. The number of original features resulting from using the aforementioned Tranalyzer tool is 107, while after implementing this stage; the number of features becomes 35 features. Then, we apply feature extraction to the selected features to add 40 other features to become 75 features in total which are named as compound features. The distribution of ISOT- CID dataset is highly imbalanced. As a result, sampling techniques were employed to attain balance in the dataset. This work utilized both undersampling and oversampling methodologies to achieve equilibrium in the dataset. Undersampling was applied to classes with over 200,000 examples, while oversampling was used for classes with fewer than 200,000 examples, aiming to balance the data in both the training and test sets. In the training set, undersampling was applied to all classes with over 200,000 examples, while oversampling was applied to classes with fewer than 200,000 examples. Similarly, in the test set, oversampling was applied to classes by 70,000 examples. Finally, the training set consist of 1,600,000 examples whereas the testing set consist of 560,000 examples.

The IOST-CID dataset is fed into the machine learning algorithm at the preprocessing step. The preprocessing step is composed of two steps: data transformation and normalization to prepare the data in a useful format to be used in the machine learning model. The sampling method is used after the processing step for solving the problem of imbalanced classes. The training dataset subsequently serves to train three different machine learning models: MLP, KNN, and SVM. Finally, the three models are tested by applying them to the testing data to run the classification process.

FEATURES EMPLOYED

Aldribi et al. (2020a), Shows that can extract 3 different type of features: frequency features, entropy features, and load features. These features are computed using an approach known as the Reinmann chunking scheme, which involves dividing the flow based on time windows to extract the features. The row features used to extract frequency, entropy, and load features to compute these new features by Riemann chunking scheme are: Source port, Number of packets sent, Destination port, Number of packets received, Source MAC and Destination MAC. By applying these techniques, we extract 16 frequency features, 16 entropy features, and 8 load features.

Frequency Features

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Performing many experiments with different time window sizes to extract features from the data we found that a window equal to 100 is the best hyperparameter. Lu and Traore (2005), computed frequency features for intrusion detection of the conventional networks. Actually, only features of the incoming traffic that reach a particular destination without looking at their original places are computed. For instance, the outgoing traffic in the cloud paradigm is very essential for detection anomalies. Moreover, malicious traffic may arise due to source abstraction when calculating frequency features. This occurs because the malicious traffic is concealed within legitimate traffic originating from various sources, making it difficult to detect attacks, especially when the current instance receives high-volume traffic from multiple resources. Additionally, even when such attacks are identified, tracing its source becomes significant difficulties. This challenge is compounded in scenarios where multiple threat actors coordinate attacks using different devices or machines. Furthermore, diverse frequency features are introduced to define each instance i ∈ F. The detailed frequency features shows in Table II. The employed frequency features definition can be found in this study Aldribi et al. (2020a).

Entropy Features

Entropy analysis plays a fundamental role in attack classification frameworks, as it enhances the performance of machine learning techniques while also offering statistical benchmarks as mentioned in Aldribi et al. (2020a). If the fixed extracted feature f has positive values of a parameter j1, . . ., jk , computing of the entropy of this feature will be as follows: Let f = v ji , i = 1, . . ., k be the set of possible values of f, v ji , for each value of a parameter, ji , in which the interval of observation time is [t, t + delta t]. The entropy of f with respect to j is:

$$H_f(t) = -\sum_{i=1}^{k} \left(\frac{v_{ji}}{v}\right) \log_2 \left(\frac{v_{ji}}{v}\right) \qquad (1)$$

where v = ∑ (i = 1) k=1 v (j i) is the sum of values in f. The entropy value lies within the range of [0, log 2 k]. When a feature's behavior is deterministic, its entropy value is 0. On the other hand, when the feature shows complete randomness, its entropy has the maximum value of log 2 k. In certain cases, the entropy may depend on other parameters. Upon fixing and implementing these parameters, the conditional entropy measure may be derived for specific instances of the predetermined parameters. In Lu and Traore (2005) paper can be computed as particular cases of our proposed models (KNN,SVM,MLP) , which is more general. As an example, the entropies f (s , i) i n , f (sp , i) i n, and f (i , ip) i n with respect to s, Source Port (sp) and Internet Protocol (ip), respectively, as follows:

$$H_i\left(f^{in}_{s,i}, t\right) = -\sum_{s} \left(\frac{f^{in}_{s,i}}{\sum_s f^{in}_{s,i}}\right) \log_2 \left(\frac{f^{in}_{s,i}}{\sum_s f^{in}_{s,i}}\right) \qquad (2)$$

$$H_i\left(f^{in}_{sp,i}, t\right) = -\sum_{sp} \left(\frac{f^{in}_{sp,i}}{\sum_{sp} f^{in}_{sp,i}}\right) \log_2 \left(\frac{f^{in}_{sp,i}}{\sum_{sp} f^{in}_{sp,i}}\right) \qquad (3)$$

$$H_i\left(f^{in}_{i,ip}, t\right) = -\sum_{ip} \left(\frac{f^{in}_{i,ip}}{\sum_{ip} f^{in}_{i,ip}}\right) \log_2 \left(\frac{f^{in}_{ip,i}}{\sum_{ip} f^{in}_{i,ip}}\right). \qquad (4)$$

Load Features

The load features for a network flow can be computed by the following equation: dividing the number of packets which are incoming to the number of packets which are outgoing as follows:

$$L_{j,j'}(t) = \frac{f^{in}_{j}(t)}{f^{out}_{j'}(t)} \qquad (5)$$

where j and j' are defined as subscripts of the corresponding in and out frequency features.

Computing the Features

There are several ways to group packets into flows that produce meaningful statistical observations. In this paper, a Riemann scheme-based Aldribi et al. (2020a) approach is used to compute features by using time as window. The

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

features can be categorized according to how the integral is defined and calculated in the Riemann sense as reflected in Figure 4. Moreover, time is x axis that is divided into windows disjoint delta t windows. Then, the observations included inside each window are the tools for computing the features. The Riemann chunking scheme is the expression used to refer to this approach.

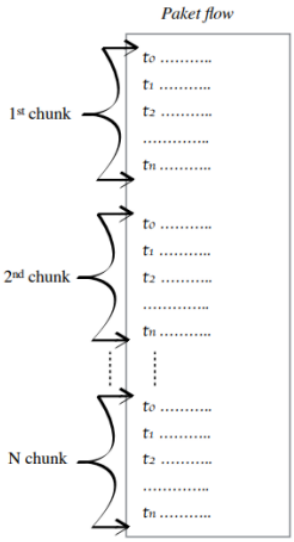

Figure 4. Riemann Chunking SchemeAldribi et al. (2020a)

Table II Description of Frequency Features

Feature	meaning
Frequency features	
frq_in_to_ed	The number of packets flowing from the endpoint es to ei during [t, t + delta t] divided by delta t.
frq_in_from_allSP_ei_to_ed	The number of packets flowing from all sp in the end-point es to ei during [t, t + delta t] divided by delta t.
frq_in_from_ei_to_allIP_ed	The number of packets flowing from the endpoint es to all ip in ei during [t, t + delta t] divided by delta t.
frq_in_from_allSP_ei_to_allIP_ed	The number of packets flowing from all sp in the end-point es to all ip in ei during [t, t + delta t] divided by delta t.
frq_in_from_SP_allei_to_allIP_ed	The number of packets flowing from specific sp in all endpoint's es to all ip in ei during [t, t + delta t] divided by delta t.
frq_in_from_allSPall_ei_to_ed	The number of packets flowing from all sp in all end-points es to specific ip in ei during [t, t + delta t] divided by delta t.
frq_in_to_ei	The total number of packets flowing to ei during [t, t + delta t] divided by delta t
max_numPkt_over_IP_to_ei	The maximum number of packets over ip flowing to ei during [t, t + delta t] divided by delta t
frq_out_to_ei	The number of packets flowing from the endpoint ei to ed during [t, t + delta t] divided by delta t.
frq_out_from_allSP_es_to_ei	The number of packets flowing to all d p from the endpoint ei to ed during [t, t + delta t] divided by delta t.
frq_out_from_es_to_allIP_ei	The number of packets flowing from all ip in the end-point ei to ed during [t, t + delta t] divided by delta t.
frq_out_from_allSP_es_to_allIP_ei	The number of packets flowing from all ip in the end-point ei to all d p in ed during [t, t + delta t] divided by delta t.
frq_out_from_SP_allei_to_allIP_es	The number of packets flowing from all ip in ei to specific d p in all endpoints ed during [t, t + delta t] divided by delta t.
frq_out_from_allSPall_es_to_ei	The number of packets flowing from specific ip in ei to all d p in all endpoints ed during [t, t + delta t] divided by delta t.
frq_out_from_ei	The total number of packets flowing from ei during [t, t + delta t] divided by delta t.
max_numPkt_over_IP_from_ei	The maximum number of packets over ip flowing out of ei during [t,t + delta t] divided by delta t.

REDUCTION OF DIMENSIONALITY

To reduce the effect of features that contribute less to the classification process, principal component analysis (PCA) is used as described in Pearson (1901). PCA helps to reduce the dimensionality of the feature space, thus reducing the number of features required for further processing steps. Actually, PCA is a nonparametric method that is

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

distinguished by its ability to automatically define the significant input feature and by transform the features into new dimensions and take the best dimensions to describe the data. For each flow, only a specific subset of the previously defined feature vector components will be used as the output for the detection algorithm. This is determined through exploratory experiments conducted on a sample of the data, it was determined that the first four features, as obtained by PCA, are retained (cutoff) and outputted for further analysis. Actually, the final form of the representations of data obtained after carrying out PCA takes the shape of a sequence of observations that consists of a stream of X1 , X 2 ,...XN because the remaining PCA features take zero value. Finally, every observation parallels an optimized feature vector Xi that represents a separate packet flow.

PROPOSED ML PIPELINE APPROACH

After raw packet attributes are reduced to hypervisor- and instance-level measures, CloudIDS needs to choose when to pass those observations over to its change-point detectors. Two windowing techniques are employed, one optimized for two different traffic behavior patterns seen within the ISOT-CID experience.

Data Pipline

Figure 5 shows the full pipeline of the data and machine learning models. First, we have the ISOT-CID dataset and apply some data cleaning techniques like remove missing data or filling it by mean or median, remove duplicate data and etc. After that, we compute the correlation between the features and itself to remove highly correlated features and apply the correlation between the output and the features to select the best features in feature selection step. Then, we extract new features by get the frequency features, entropy features, and load features that we discussed earlier. Finally, we have an optional step is reduce dimensionality of the data by using PCA to overcome the problem of high inference time. The next step is data balancing to avoid bias problem.

Machine Learning Pipeline

The proposed machine learning models is represented in Figure 5 as we discussed earlier we have three models: MLP, KNN, and SVM. After splitting the data into training and testing we train three models and measure the performance of every model on testing data then get the best model to use in production.

Model Deployed on cloud Network

As shown in Figure 6 the model check the incoming packets and provide the necessary information to the firewall to take the action. The model pass the benign packets only and reject any other packets that classified as attack.

Figure 5. Proposed Machine Learning Classification

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

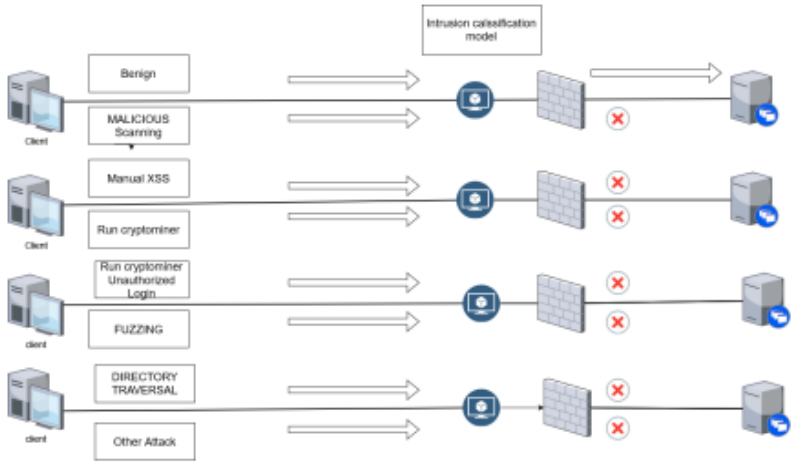

Figure 6. Intrusion Classification Model deployed on the cloud

EXPERIMENTS AND RESULTS

In this paper, the proposed framework is developed, implemented, and tested the aforementioned three ML-based approaches and employ the classification models KNN, SVM, and MLP learning techniques. The original ISOT-CID dataset is used and feature extracting techniques are applied to increase the features. The model evaluation is an essential part of selecting the best machine learning model for classification. Evaluation metrics are used to discriminate between the results of different models. To measure the performance of the proposed models, we used metrics such as accuracy and F1-score.

True Positives (TP) measure the number of instances in the positive class that the model correctly identifies as positive.

True negatives (TN) measure the number of instances in the negative class that the model correctly identifies as negative.

False positives (FP) measure the number of instances in the which are wrongly identified as positive class.

False negatives (FN) measure the number of instances in the which are wrongly identified as negative class.

The metrics used for evaluation are as follows:

Accuracy: Accuracy defines the percentage of correct predictions of the model and can be given as:

$$\text{Accuracy} = \frac{TN + TP}{TN + TP + FN + FP} \qquad (6)$$

F1-Measure: F1 measure provides the harmonic mean of recall and precision. F1 ranges from 0 to 1 where 0 represents a model that fails at prediction and 1 is a representative of a model that is perfect.

$$\text{F1 measure} = \frac{2TP}{FN + FP + 2TP} \qquad (7)$$

The classification task works by assigning a label to each sample in the dataset. The dataset has 35 numeric features. Once preprocessing is done, the dataset is prepared, and the next step involves feeding the data into machine learning models to train the network and find the best model. In our case, the problem is multiclass classification, where K-Nearest Neighbors, Support Vector Machine, and Multi-Layer Perceptron algorithms are employed to classify the types of attacks. The accurate classification process requires us to select the best models, depending on the parameters that are selected and that give the best results. We utilize the F1-score and accuracy to evaluate and compare the models. This evaluation framework incorporated four different experimental configurations to examine the impact of incorporating additional features into the system

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Evaluation Results of MLP

The main objective of our proposed Multi-layer Perceptron (MLP) model was to measure its performance in classifying intrusions for multiclassification. The model's performance was assessed utilizing the ISOT-CID dataset, which was divided into testing and training data without any preprocessing. The proposed model was configured with various parameters to improve its overall classification capability. To find the optimal hyperparameters, we first employed the early stopping approach that identifies when the network should stop training based on the validation loss. We set a patience value of three for early stopping. For activation functions, we used Tanh for hidden layers and SoftMax for the last layer. This is because it consists of more than one neuron. Additionally, we utilized the Adam optimizer with different learning rate values and categorical cross-entropy as the loss function. We also experimented with various batch sizes, learning rates, layer sizes, and the number of nodes in each layer to optimize the model. We focused on parameters such as layer size and the number of neurons within each layer to get the best model before feature extraction.

The results of our experiments related to the setting Pre Feature Extraction without PCA and After Feature Extraction without PCA was reported in Alamer and Aldribi (2022).

Before Feature Extraction with PCA (10 components)

We apply PCA to the 35 original features and then get the best 10 features that passes it to the model. Figure 7 and Figure 8 clarify the accuracy and loss over epochs.

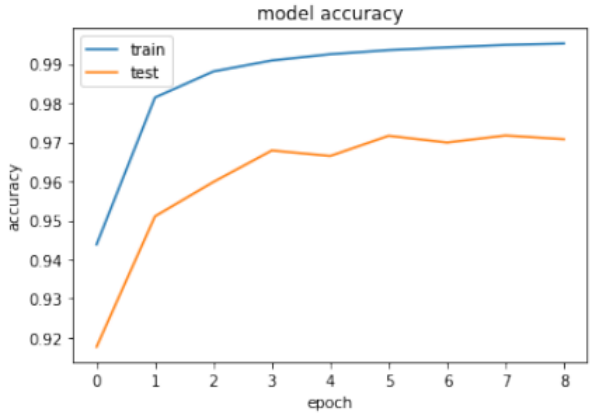

Figure 7. Model Accuracy Over Epochs Before Feature Extrac-

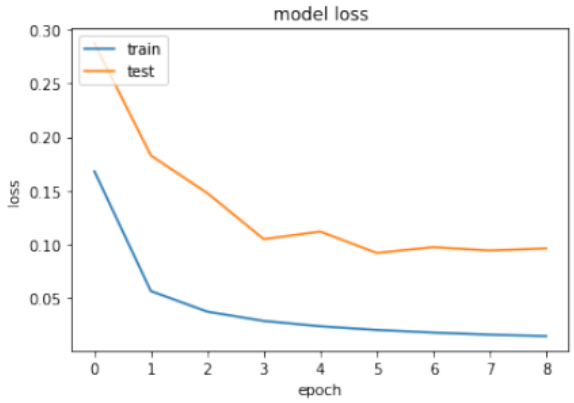

Figure 8. Model Accuracy and Loss Over Epochs Before Feature

After Feature Extraction with PCA (20 components)

We apply a feature model that generates 40 new features added to 35 features so we have 75 total features. Then, we

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

apply PCA to get the best 20 features (the number 20 is getting by keep the model performance as long as decrease the dimensions) and then passes it to the model to get the results. In Figure 9 and Figure 10 clarify the accuracy and loss over epochs. We found the model after 1 epoch is decreasing in accuracy so stop training at epoch number 1. The developed MLP technique is tested based on different metrics such as F1-measure and accuracy. Table III shows that we have 4 different results in different scenarios of data (original data, data with PCA, data with extracted features, and data with extracted features and PCA). The MLP model performs well in the dataset. As the number of features used is increased, the classification accuracy is improved. The best performing model is the MLP, even better than when PCA is used on the data. When using PCA, the goal is to identify the optimal dimensions to represent the data while losing information. As a result, MLP with extracted features is the best experiment and the confusion matrix of the classes are shown in Figure 11.

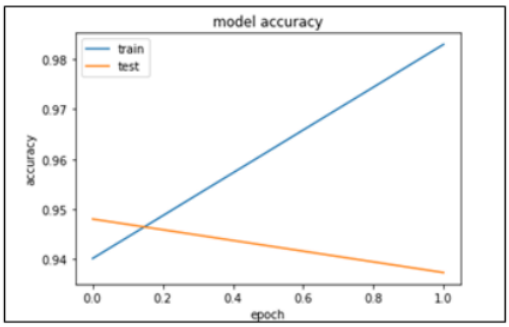

Figure 9. Model Accuracy Over Epochs After Feature Extraction

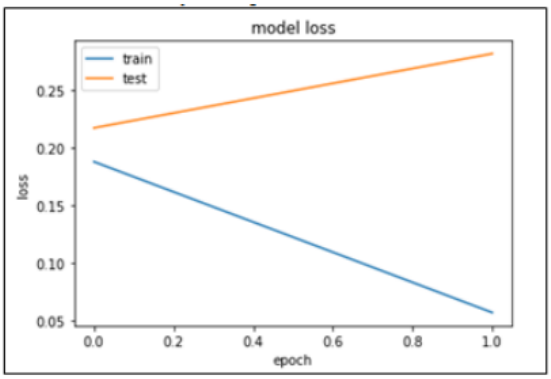

Figure 10. Model Loss Over Epochs After Feature Extraction and

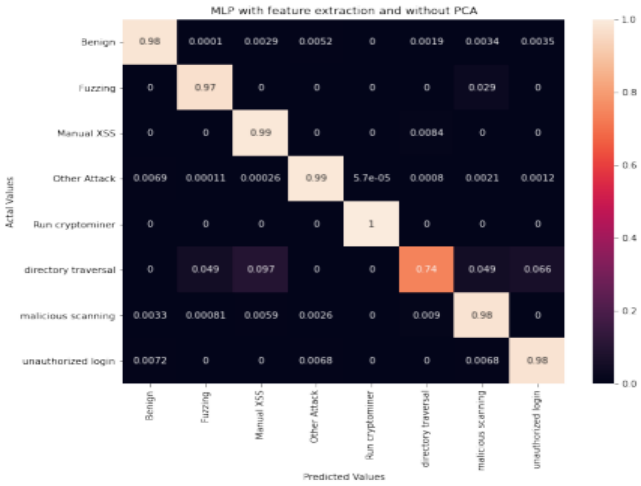

Figure 11. Confusion Matrix of MLP with Features

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table III Evaluation Metrics of Different Classes for the MLP Model

Model	Accuracy	F1-score
MLP with feature extraction and without PCA	**95.28%**	**95.26**
MLP with feature extraction and with PCA	94.8%	94.89%
MLP without feature extraction and PCA	93.9%	94%
MLP without feature extraction and with PCA	90%	90%

Performance of KNN

The second approach was KNN, which was trained on IOST-CID dataset and evaluated for multiple attack classification. To obtain the best hyperparameters for KNN, we utilized hyperparameters optimization (on 10% of original data to speed the process of choosing the best hyperparameters) with different parameters. when k close to 1 the model goes to overfitting zone and when k is large the model goes to unfitting zone and the result shows that this collection of parameters is considered the best 'metric ': 'Manhattan ', 'n neighbors ': 8, 'weights ': uniform before and after feature extraction these hyperparameters give the best accuracy. Table IV shows evaluation of KNN in different scenarios. KNN is one of the simplest models that have a good performance with original data (without any preprocessing) but, it will perform better if we have more features as in the experiments, and as we discussed previously, PCA is used to reduce dimensionality and reduce the overall performance of the model. Therefore, for all 4 KNN experiments, the KNN with features performs the best and the confusion matrix of the classes are shown in Fig.12.

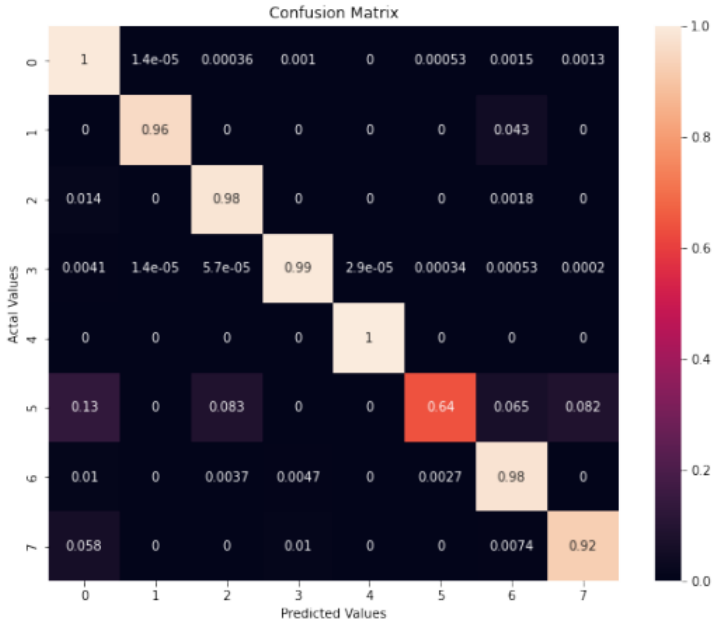

Figure 12. Confusion Matrix of KNN with Features

Table IV Evaluation Metrics of Different Classes for the KNN Mode

Model	Accuracy	F1-score
KNN with feature extraction and without PCA	**93.4%**	**93.1%**
KNN with feature extraction and with PCA	90.9%	90.45%
KNN without feature extraction and PCA	92.5%	92.1%
KNN without feature extraction and with PCA	87.4%	86.9%

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Performance of SVM

The third model was SVM, which is trained on ISOT-CID dataset for classification and evaluated for multiple attack classification, to obtain the best hyperparameters for SVM. We tried on polynomial and linear kernels and the polynomial kernel goes to overfitting zone and linear kernel goes to underfittig zone. We search in regularization parameter and kernel to obtain the best accuracy we found the best hyperparameters are 'C ': 0.5, 'kernel ': 'rbf ' in addition, these hyperparameters gives the best accuracy before and after feature extraction. Table V shows evaluation of SVM in different scenarios. The SVM is unsuitable when dealing with huge data so the performance on original data (without any preprocessing) is the best in addition, when adding more features the SVM can't give more but it's much worse in addition, after PCA. The confusion matrix of the classes are shown in Figure 13.

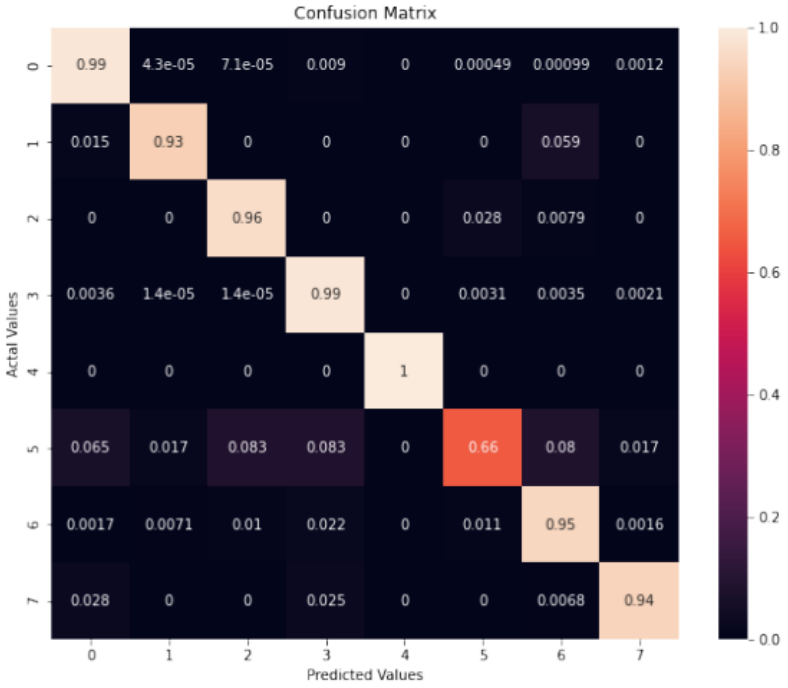

Figure 13. Confusion Matrix of SVM with Features

Table V Evaluation Metrics of Different Classes for the SVM Model

Model	Accuracy	F1-score
SVM with feature extraction and without PCA	92%	91.5%
SVM with feature extraction and with PCA	91.9%	91.7%
SVM without feature extraction and PCA	**92.5%**	**92.2%**
SVM without feature extraction and with PCA	91.8%	91.6%

ANALYSIS AND RESULTS

We conducted extensive experiments with different methods, using the ISOT-CID dataset with 1600000 instances for training and 560000 instances for testing. The three models employed the identical dataset and environment, with the model hyper parameters differing slightly. After testing, the models are tested to classify all attack classes. Based on the accuracy, we selected the optimal model performance. The following conclusions were drawn from the data:

- Accuracy is the first statistic employed to evaluate the effectiveness of the proposed frameworks. The MLP model has the greatest testing accuracy rate of 95.28%, compared to 92.5% and 93.4% for the SVM and KNN models, respectively.

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

- The f1-score is the second statistic, which is important when dealing with an unbalanced dataset. The f1-score calculated for the three models MLP, SVM, and KNN. MLP scored is 95.26%, While the score for SVM is 92.2%, and 93.1% for the KNN, indicating that the MLP model outperformed both the SVM and the KNN models.
- When we compared the performance of the MLP model to that of the KNN and SVM models, we found that MLP produced better results for all of the metric values.
- The MLP model properly classified the majority of malicious scanning, manual XSS, run crypto miner, unauthorized login, fuzzing, directory traversal, ping, and other threats as attack classes.
- Compared the four experiments of MLP as a summary of our findings, Table VI and Fig. 14 presenting all proposed approaches evaluation results, and the MLP is the model that outperforms the other two models. Moreover, shows in Table VII the comparison between our models and the state of the art.

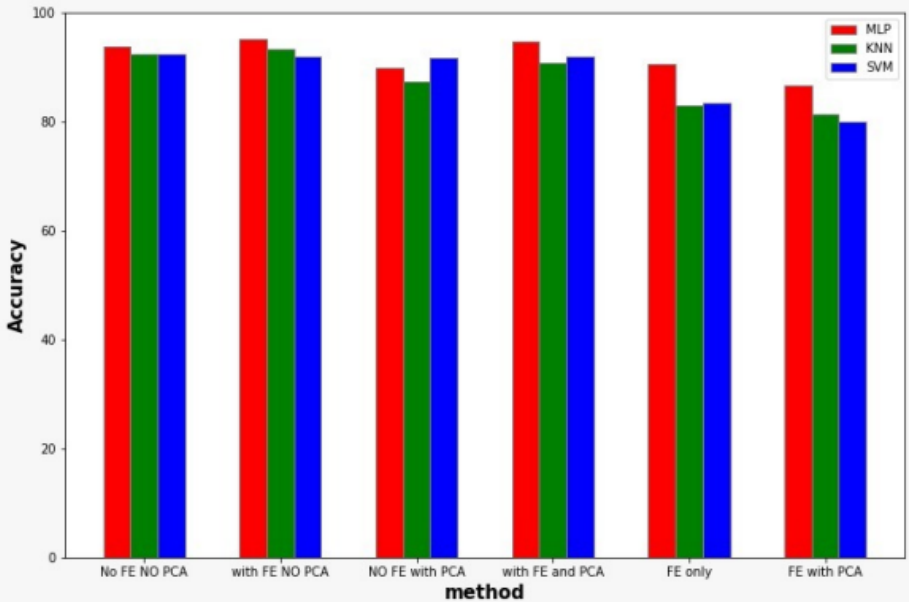

Figure 14. Proposed Approaches Evaluation Results

Table VI Evaluation Metrics of Different Classes for the SVM Model

Model	Accuracy	F1-score
MLP without feature extraction and PCA	93.9%	94%
MLP with feature extraction and without PCA	**95.28%**	**95.26%**
MLP without feature extraction and with PCA	90%	90%
MLP with feature extraction and with PCA	94.8%	94.89%
MLP with extracted feature and without PCA	90.7%	90.6%
MLP with extracted feature and with PCA	86.6%	86.5%
KNN without feature extraction and PCA	92.5%	92.1%
KNN with feature extraction and without PCA	**93.4%**	**93.1%**
KNN without feature extraction and with PCA	87.4%	86.9%
KNN with feature extraction and with PCA	90.9%	90.45%
KNN with feature extracted and without PCA	82.9%	82.3%
KNN with feature extracted and with PCA	80%	80%
SVM without feature extraction and PCA	**92.5%**	**92.2%**
SVM with feature extraction and without PCA	92%	91.5%
SVM without feature extraction and with PCA	91.8%	91.6%
SVM with feature extraction and with PCA	91.9%	91.7%
SVM with feature extracted and without PCA	83.4%	83.4%
SVM with feature extracted and with PCA	81.4%	82%

XIII. Analysis and Results

Journal of Information Systems Engineering and Management
2025, 10(43s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

	Paper	Year	Targeted environment	Dataset	Preprocessing techniques	Model type	Model accuracy
1	Firdausi, Lim, Erwin, and Nugroho [6]	2010	VM	TERM FREQUENCY	feature selection	SVM	91.7% (Detection)
						KNN	92.9% (Detection)
						MLP	94.2% (Detection)
2	Kannan, Maguire, Sharma, and Schoo [9]	2012	Cloud Hypervisor	KDD Cup 99	feature selection	SVM	83.58% (Detection)
						FSVM	92.46% (Detection)
						FSVM	98.51% (Detection)
3	Pandeeswari and Kumar [11]	2015	Cloud Hypervisor	KDD cup 1999	-	FCM-ANN	98% (Detection)
4	Ravale, Marathe, and Padiya [27]	2015	Not mentioned	KDDCUP'99	-	KMSVM	90.97% (Detection)
						KM	78.23% (Detection)
						SVM	59.11% (Detection)
5	Mighan and Kahani [10]	2018	Not mentioned	ISCX IDS UNB 2012	normalization	PCA-SVM	85.6% (Detection)
						DL-SVM	90.2% (Detection)
6	Jamadar [22]	2018	Not mentioned	CICIDS 2017	Feature selection using RFE	DTREE	99% (Detection)
7	Aldribi, Traoré, Moa, and Nwamuo [19]	2020	Cloud Hypervisor	ISOT-CID	-	multivariate statistical change analysis	96.23% (Detection)
8	Alshammari, Aldribi [36]	2021	Cloud Hypervisor	ISOT-CID	Feature extraction	DTREE	100% (Detection)
					Feature extraction	MLP	94% (Detection)
9	Our paper	2022	Cloud Hypervisor	ISOT-CID	Feature extraction only	MLP	95.28% (Detection + Classification)
					Feature extraction and PCA	MLP	93.9% (Detection + Classification)

This work shares similarities with the study by Alshammari and Aldribi (2021) in terms of utilizing the ISOT-CID dataset for model training, however, the objectives differ. While their research focuses solely on attack detection, our approach extends to both detection and classification. Consequently, our approach achieves a performance accuracy of approximately 95%, while Amirah's model reaches around 100% accuracy. Additionally, the studies by Chou and Wang (2016), Modi et al. (2012), Sakr et al. (2019), Ravale et al. (2015), and Pandeeswari and Kumar (2016) differ from our work in that they do not reflect the actual cloud network environment, which contrasts with conventional network architectures in terms of both structure and network flow capacity. Regarding the models used, Denning (1987) employ a rule-based system, while Modi et al. (2012) use a Bayesian classification algorithm for network anomaly detection.

Kannan et al. (2012) proposed an IDS based on combining Genetic-based feature selection algorithm and Fuzzy Support Vector Machines (FSVM). The work in Chou and Wang (2016) utilized unsupervised learning algorithm for constructing a model of decision trees and used it for detection anomaly detection with unlabeled data. Pandeeswari and Kumar (2016) propose hypervisor detector, It is a cloud-based anomaly detection system that uses fuzzy c-means clustering-artificial Neural Network (FCMANN).

ANALYSIS AND RESULTS

Recently, cloud computing has gained significant popularity within the IT industry and attracted attention from both government and academic sectors. Thus, there is a need to develop robust security mechanisms to address the threats faced by cloud environments. Most existing systems utilize datasets designed for traditional network architectures, which do not accurately model real-world cloud environments.

In this paper, we conducted a study to develop an intrusion classification framework for cloud computing using machine learning techniques. We employed Multilayer Perceptron (MLP), Nearest Neighbor (KNN), and Support Vector Machines (SVM) algorithms to classify anomalies in cloud network traffic. A small subset of the ISOT-CID dataset, which includes both legitimate and malicious activities, was utilized for the experiments. The experimental

framework incorporated a Riemann-based segmentation approach to systematically compute three feature classes: frequency, entropy, and load, over predefined time intervals. We identified 40 new features necessary to create a dataset suitable for training a machine learning model for intrusion classification.

When tested on a validation dataset, Multilayer Perceptrons produced the highest accuracy results. We evaluated the performance of the three classifiers (KNN, SVM, and MLP) using the newly constructed dataset. In the first evaluation step, we included all 35 attributes derived from the Tranalyzer tool. With these attributes, MLP achieved the highest accuracy of 95.28%, while SVM and KNN achieved accuracies of 92.5% and 93.4%, respectively. In the second evaluation step, we applied Principal Component Analysis (PCA) to reduce preprocessing complexity. MLP with 20 components achieved an accuracy of 94.8%, while SVM and KNN achieved accuracies of 91.9% and 90.9%, respectively, as shown in Table VII and Figure 14.

Although the performance evaluation showed positive results, the models have limitations due to long inference times. IDS security solutions for cloud networks must be extremely fast to process data in real-time, extract communication traffic features, and provide prompt responses.

The results of this study shows that the Multilayer Perceptron offers the best performance among the classifiers. Future research could be done to develop hybrid machine learning methods and improve model performance by combining different feature selection algorithms. Additionally, incorporating time series models could enhance the ability to utilize time-related features and extract more valuable insights.

REFERENCES

[1] Aldribi, A., Traore, I. & Moa, B. (2018) Data Sources and Datasets for Cloud Intrusion Detection Modeling and Evaluation, Cham, pp. 333–366. URL https://doi.org/10.1007/978-3-319-73676-1_13
[2] Aldribi, A., Traoré, I., Moa, B. & Nwamuo, O. (2020) Hypervisor-based cloud intrusion detection through online multivariate statistical change tracking. Comput. Secur., 88. doi:10.1016/j.cose.2019.101646.
[3] Aldribi, A., Traore, I., Quinan, P.G. & Nwamuo, O. (2020) Documentation for the isot cloud intrusion detection benchmark.
[4] Alshammari, A. & Aldribi, A. (2021) Apply machine learning techniques to detect malicious network traffic in cloud computing.
[5] Journal of Big Data, 8(1), 90. doi:10.1186/s40537-021-00475-1.URL https://doi.org/10.1186/s40537-021-00475-1
[6] Catteddu, D. & Hogben, G. (2009) Cloud computing security risk assessment. Enisa, no„ 1–2.
[7] Chou, H.H. & Wang, S.D. An adaptive network intrusion detection approach for the cloud environment. In: Conf, C. (Ed.) Proc.Int,
[8] 2016. vol. 2015-Janua: Secur. Technol.
[9] Chumachenko, K. Machine learning methods for malware detection and classification. In: Proc. 21st Pan-Hellenic Conf. Informatics PCI 2017, 2017. : p. 93.
[10] Denning, D.E. (1987) An intrusion detection model. no., 2, 222-232.
[11] Dhanabal, L. & Shantharajah, S.P. (2015) A study on nsl-kdd dataset for intrusion detection system based on classification algorithms. Int. J. Adv, 4(6), 446–452. doi:10.17148/IJARCCE.2015.4696.
[12] Firdausi, I., Lim, C., Erwin, A. & Nugroho, A.S. Analysis of machine learning techniques used in behavior-based malware detection.In: Proc. 2010 2nd Int, 2010. no. December: Conf. Adv. Comput.Control Telecommun. Technol. ACT 2010, pp. 201–203.
[13] Ikram, S.T. & Cherukuri, A.K. (2016) Improving accuracy of intrusion detection model using pca and optimized svm. J. Comput.Inf, 24(2), 133–148. doi:10.20532/cit.2016.1002701.
[14] Jamadar, R.A. (2018) Network intrusion detection system using machine learning. Indian J. Sci, 11(48), 1–6. doi:10.17485/ijst/2018/v11i48/139802.
[15] Kannan, A., Maguire, G.Q., Sharma, A. & Schoo, P. Genetic algorithm based feature selection algorithm for effective intrusion detection in cloud networks. In: Proc. 12th IEEE Int, 2012. no. January: Conf. Data Min. Work. ICDMW 2012, pp. 416-423.
[16] Khan, Y., Jan, Z.H. & Nagaveni, V. (2019) Smart intrusion detection using machine learning techniques

Research Article

modified19 jan2019 citation.vol., 7, 15 – –20.
[17] Lu, W. & Traore, I. (2005) A new unsupervised anomaly detection framework for detecting network attacks in realtime. Lect. Notes Comput. Sci, 3810, 96–109. doi:10.1007/115993719.
[18] Mcculloch, W.S. & Pitts, W. (1990) A logical calculus nervous activity. Bull. Math, 52, 99–115.
[19] Mehmood, Y., Shibli, M.A., Habiba, U. & Masood, R. Intrusion detection system in cloud computing: Challenges and opportunities. In: Conf. Proc. 2013 2nd Natl, 2013. no. December: Conf. Inf. Assur.NCIA 2013, pp. 59–66.
[20] Mighan, S.N. & Kahani, M. (2018) Deep learning based latent feature extraction for intrusion detection, 26, 1511–1516. doi:10.1109/ICEE.2018.8472418.
[21] Modi, C.N. & Patel, D. A novel hybrid-network intrusion detection system (h-nids) in cloud computing. In: Proc. 2013 IEEE Symp, 2013. pp.23–30: Comput. Intell. Cyber Secur. CICS IEEE Symp. Ser. Comput.Intell. SSCI 2013, pp. 2013–2013.
[22] Modi, C.N., Patel, D.R., Patel, A. & Muttukrishnan, R. (2012) Bayesian classifier and snort based network intrusion detection system in cloud computing, 2012, 3. doi:10.1109/ICCCNT.2012.6396086.
[23] Mulay, S.A., Devale, P.R. & Garje, G.V. (2010) Intrusiondetection system using support vector machine and decision tree. Int. J. Comput. Appl., 3(3), 40–43. doi:10.5120/758-993.
[24] Nolan, D.R. & Lally, C. (2018) An investigation of damage mechanisms in mechanobiological models of in-stent restenosis. J. Comput. Sci, 24, 132–142. doi:10.1016/j.jocs.2017.04.009.
[25] Pandeeswari, N. & Kumar, G. (2016) Anomaly detection system in cloud environment using fuzzy clustering based ann. Mob, 21(3), 494–505. doi:10.1007/s11036-015-0644-x.
[26] Pearson, K. (1901) Liii. on lines and planes of closest fit to systems of points in space. London, Edinburgh, Dublin Philos. Mag, 2(11), 559–572. doi:10.1080/14786440109462720.
[27] Ravale, U., Marathe, N. & Padiya, P. (2015) Feature selection based hybrid anomaly intrusion detection system using k means and rbf kernel function. Procedia Comput. Sci, 45, 428–435. doi:10.1016/j.procs.2015.03.174.
[28] Rosenblatt, F. (1958) The perceptron: a probabilistic model for information storage and organization in the brain. Psychol. Rev, 65(6), 386–408.
[29] Sakr, M.M., Tawfeeq, M.A. & El-Sisi, A.B. (2019) Network intrusion detection system based pso-svm for cloud computing. Int. J. Comput. Netw. Inf, 11(3), 22–29. doi:10.5815/ijcnis.2019.03.04.
[30] Salo, F., Nassif, A.B. & Essex, A. (2019) Dimensionality reduction with ig-pca and ensemble classifier for network intrusion detection. Comput. Networks, 148, 164–175. doi:10.1016/j.comnet.2018.11.010.
[31] Subba, B., Biswas, S. & Karmakar, S. (2016) A neural network based system for intrusion detection and attack classification, 2016, 22. doi:10.1109/NCC.2016.7561088.
[32] tranalyzer Development Team (2023) tranalyzer: High-performance network traffic analyzer. http://tranalyzer.com/, accessed: [Insert Last Accessed Date].
[33] Verwoerd, T. & Hunt, R. (2002) Intrusion detection techniques and approaches. Comput. Commun., 25(15), 1356–1365. doi:10.1016/S0140-3664(02)00037-3.
[34] Alamer, K. and Aldribi, A., 2022, December. Intrusion Classification for Cloud Computing Network: A Step Towards an Intelligent Classification System. In 2022 14th International Conference on Computational Intelligence and Communication Networks (CICN) (pp. 757-764). IEEE.