

# ThreatSense: Intelligent Threat Detection

by

Sai Charan Annam (0964317)

Vani Janagani (0943573)

Pavan Kalyan Pathi (0946714)

Submitted in partial fulfillment  
of the requirements for the degree of  
Master of Science in Cybersecurity



**Sacred Heart  
UNIVERSITY**

---

SCHOOL OF COMPUTER SCIENCE & ENGINEERING

---

School of Computer Science and Engineering  
Sacred Heart University

Supervised by  
Dr. Sajal Bhatia  
November 26, 2025

# Abstract

One of the major challenges in cyber security is the provision of an automated and effective cyber-threats detection technique. In this paper, we present an AI technique for cyber-threats detection, based on artificial neural networks. The proposed technique converts multitude of collected security events to individual event profiles and use a deep learning-based detection method for enhanced cyber-threat detection. For this work, we developed an AI-SIEM system based on a combination of event profiling for data preprocessing and different artificial neural network methods, including FCNN, CNN, and LSTM. The system focuses on discriminating between true positive and false positive alerts, thus helping security analysts to rapidly respond to cyber threats. All experiments in this study are performed by authors using two benchmark datasets (NSLKDD and CICIDS2017) and two datasets collected in the real world. To evaluate the performance comparison with existing methods, we conducted experiments using the five conventional machine-learning methods (SVM, k-NN, RF, NB, and DT). Consequently, the experimental results of this study ensure that our proposed methods are capable of being employed as learning-based models for network intrusion-detection, and show that although it is employed in the real world, the performance outperforms the conventional machine-learning methods.

# Acknowledgement

First of all, we extend our sincere gratitude to our project guide, Associate Professor Dr. Sajal Bhatia, from the School of Computer Science and Engineering, Sacred Heart University, for giving us the opportunity to work on this project. His guidance, encouragement, and valuable insights have been helpful in shaping the direction and quality of the work.

This project has significantly enhanced our technical skills, deepened our understanding of programming and machine learning concepts, and improved the way we approach and execute real-world research problems. It has also strengthened our presentation, analytical, and interpersonal skills, which will continue to benefit us in our academic and professional journey.

We also express our appreciation to the faculty members of the Computer Science and Cybersecurity Department for their support and for always being willing to assist whenever needed. Their mentorship has contributed greatly to the successful completion of this project.

# Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
<b>2</b>	<b>Background Research and Literature Survey</b>	<b>9</b>
2.1	Background Research . . . . .	9
2.2	Existing System . . . . .	9
2.3	Literature Survey . . . . .	10
2.3.1	Enhanced Network Anomaly Detection based on Deep Neural Networks . . . . .	10
2.3.2	Network Intrusion Detection based on Directed Acyclic Graph and Belief Rule based . . . . .	11
2.4	Preliminaries . . . . .	11
2.4.1	IDS / IPS and SIEM . . . . .	11
<b>3</b>	<b>Proposed System and System Architecture</b>	<b>13</b>
3.1	Proposed System . . . . .	13
3.1.1	Advantages . . . . .	13
3.2	System Requirement Analysis . . . . .	14
3.2.1	Feasibility Study . . . . .	14
3.3	Data Labeling for Learning . . . . .	15
3.4	Datasets . . . . .	16
3.4.1	NSLKDD . . . . .	16
3.4.2	CICIDS 2017 . . . . .	16
3.5	System Architecture . . . . .	16
<b>4</b>	<b>Models and Testing Types</b>	<b>19</b>
4.1	Model Selection . . . . .	19
4.1.1	Random Forest . . . . .	19
4.1.2	K-Nearest Neighbors (KNN) . . . . .	20
4.1.3	Support Vector Machine (SVM) . . . . .	20
4.1.4	Convolutional Neural Network (CNN) . . . . .	21
4.1.5	Decision Tree . . . . .	21
4.1.6	Naive Bayes . . . . .	22
4.2	Testing and Types of Testing being performed . . . . .	22
4.2.1	White Box Testing . . . . .	22
4.2.2	Black Box Testing . . . . .	22
4.2.3	Integration Testing . . . . .	23
4.2.4	Test Strategy and Approach . . . . .	23
4.3	Performance Evaluation Results and Discussion . . . . .	23

<b>5</b>	<b>Conclusion and Future Work</b>	<b>25</b>
5.1	Conclusion . . . . .	25
5.2	Future Work . . . . .	25
	<b>Bibliography</b>	<b>28</b>

# List of Figures

3.1	System Architecture . . . . .	17
4.1	Random Forest . . . . .	19
4.2	K-Nearest Neighbors . . . . .	20
4.3	Support Vector Machine . . . . .	20
4.4	Convolutional Neural Network . . . . .	21
4.5	Decision Tree . . . . .	21
4.6	Naive Bayes . . . . .	22

# List of Tables

4.1 Performance Evaluation Metrics of All Models . . . . . 24

# Chapter 1

## Introduction

With the emergence of artificial intelligence (AI) techniques, learning-based approaches for detecting cyber attacks, have become further improved, and they have achieved significant results in many studies. However, owing to constantly evolving cyber attacks, it is still highly challenging to protect IT systems against threats and malicious behaviors in networks. Because of various network intrusions and malicious activities, effective defenses and security considerations were given high priority for finding reliable solutions.

Traditionally, there are two primary systems for detecting cyber-threats and network intrusions. An intrusion prevention system (IPS) is installed in the enterprise network and can examine the network protocols and flows with signature-based methods primarily. It generates appropriate intrusion alerts, called the security events, and reports the generating alerts to another system, such as SIEM. The security information and event management (SIEM) has been focusing on collecting and managing the alerts of IPSs. The SIEM is the most common and dependable solution among various security operations solutions to analyze the collected security events and logs. Moreover, security analysts make an effort to investigate suspicious alerts by policies and threshold, and to discover malicious behavior by analyzing correlations among events, using knowledge related to attacks.

Nevertheless, it is still difficult to recognize and detect intrusions against intelligent network attacks owing to their high false alerts and the huge amount of security data. Hence, the most recent studies in the field of intrusion detection have given increased focus to machine learning and artificial intelligence techniques for detecting attacks. Advancement in AI fields can facilitate the investigation of network intrusions by security analysts in a timely and automated manner. These learning-based approaches require to learn the attack model from historical threat data and use the trained models to detect intrusions for unknown cyber threats.

A learning-based method geared toward determining whether an attack occurred in a large amount of data can be useful to analysts who need to instantly analyze numerous events. According to, information security solutions generally fall into two categories: analyst-driven and machine learning-driven solutions. Analyst-driven solutions rely on rules determined by security experts called analysts. Meanwhile, machine learning-driven solutions used to detect rare or anomalous patterns can improve detection of new cyber threats. Nevertheless, while learning-based approaches are useful in detecting cyber attacks in systems and networks, we observed that existing learning-based approaches have four main limitations.

First, learning-based detection methods require labeled data, which enable the training of the model and evaluation of generated learning models. Furthermore, it is not straightforward to obtain such labeled data at a scale that allow accurate training of a model. Despite the need for labeled data, many commercial SIEM solutions do not maintain labeled data that can be applied to supervised learning models.

Second, most of the learning features that are theoretically used in each study are not generalized features in the real world, because they are not contained in common network security systems. Hence, it makes difficult to utilize to practical cases. Recent efforts on intrusion detection research have considered an automation approach with deep learning technologies, and performance has been evaluated using wellknown datasets like NSLKDD, CICIDS2017, and Kyoto-Honeypot. However, many previous studies used benchmark dataset, which, though accurate, are not generalizable to the real world because of the insufficient features. To overcome these limitations, an employed learning model requires to evaluate with datasets that are collected in the real world.

Third, using an anomaly-based method to detect network intrusion can help detect unknown cyber threats; whereas it can also cause a high false alert rate. Triggering many false positive alerts is extremely costly and requires a substantially large amount of effort from personnel to investigate them.

Fourth, some hackers can deliberately cover their malicious activities by slowly changing their behavior patterns. Even when appropriate learning-based models are possible, attackers constantly change their behaviors, making the detection models unsuitable. Moreover, almost all security systems have been focused on analyzing short-term network security events. To defend consistently evolving attacks, we assume that over long-term periods, analyzing the security event history associated with the generation of events can be one way of detecting the malicious behavior of cyber attacks.

These challenges form the primary motivation for this work. To address these challenges, we present an AI-SIEM system which is able to discriminate between true alerts and false alerts based on deep learning techniques. Our proposed system can help security analysts rapidly to respond cyber threats, dispersed across a large amount of security events. For this, the proposed the AI-SIEM system particularly includes an event pattern extraction method by aggregating together events with a concurrency feature and correlating between event sets in collected data. Our event profiles have the potential to provide concise input data for various deep neural networks. Moreover, it enables the analyst to handle all the data promptly and efficiently by comparison with long term history data.

# Chapter 2

## Background Research and Literature Survey

### 2.1 Background Research

Due to the rapid expansion of network-linked systems, online services, and data sharing, cybersecurity has become one of the most important issues of the current digital infrastructure. As organizations adopt increasing reliance on interconnected systems, the complexity, frequency, and scale of cyber threats - such as malware, ransomware, phishing, and advanced persistent threat (APTs) - has increased alongside its reliance on a number of systems. Conventional security systems, including rule-based intrusion detection systems (IDS) and signature-based firewalls, often cannot detect sophisticated and evolving cyber attacks because these systems strictly utilize pre-defined attack patterns and set behaviour of lines. This resulted in the inability of traditional systems to detect attacks that have no prior signature (zero-day) and behaviors are classified as an intrusion.

This can contribute to systems returning high rates of positives, or false positives, while contemplating detection time. To counter the restrictions of existing systems, researchers have been able to provide a new prospect for genuine ongoing, planetary diversified threat detection systems that rely on artificial intelligence (AI) and the subsequent methodology of machine learning (ML). Statistical models (e.g., Support Vector Machine (SVM), Random Forests (RF), Decision Trees (DT), k-Nearest Neighbors (k-NN), and Naïve Bayes (NB)) capable of providing accurate information about intrusions have demonstrated a great degree of success in detection and characterize the known signs. At the same time, they all depend on pre-learned instances of historical data and therefore lack the capacity to recognize both temporal and spatial relationships present within network traffic data leading to underperformance once they encounter a burst of classes containing action. [1]

### 2.2 Existing System

Traditionally, there are two primary systems for detecting cyber-threats and network intrusions. An intrusion prevention system (IPS) is installed in the enterprise network, and can examine the network protocols and flows with signature-based methods primarily. It generates appropriate intrusion alerts, called the security

events, and reports the generating alerts to another system, such as SIEM. The security information and event management (SIEM) has been focusing on collecting and managing the alerts of IPSs. The SIEM is the most common and dependable solution among various security operations solutions to analyze the collected security events and logs. Moreover, security analysts make an effort to investigate suspicious alerts by policies and threshold, and to discover malicious behavior by analyzing correlations among events, using knowledge related to attacks.

### Disadvantages

- It is still difficult to recognize and detect intrusions against intelligent network attacks owing to their high false alerts and the huge amount of security data.
- These learning-based approaches require to learn the attack model from historical threat data and use the trained models to detect intrusions for unknown cyber threats.
- False Positives : Signature-based methods can generate a high number of false positives, where benign activities are incorrectly flagged as malicious. This can overwhelm security teams and reduce overall efficiency.
- Limited Visibility: IPS focuses on network traffic and may miss threats that originate from within the network, such as those arising from endpoints or internal users. [2]

## 2.3 Literature Survey

### 2.3.1 Enhanced Network Anomaly Detection based on Deep Neural Networks

Due to the monumental growth of Internet applications in the last decade, the need for security of information network has increased manifold. As a primary defense of network infrastructure, an intrusion detection system is expected to adapt to dynamically changing threat landscape. Many supervised and unsupervised techniques have been devised by researchers from the discipline of machine learning and data mining to achieve reliable detection of anomalies. Deep learning is an area of machine learning which applies neuron-like structure for learning tasks. Deep learning has profoundly changed the way we approach learning tasks by delivering monumental progress in different disciplines like speech processing, computer vision, and natural language processing to name a few. It is only relevant that this new technology must be investigated for information security applications. The aim of this paper is to investigate the suitability of deep learning approaches for anomaly-based intrusion detection system.

For this research, we developed anomaly detection models based on different deep neural network structures, including convolutional neural networks, autoencoders, and recurrent neural networks. These deep models were trained on NSLKDD training data set and evaluated on both test data sets provided by NSLKDD, namely NSLKDDTest+ and NSLKDDTest21. All experiments in this paper are performed by authors on a GPU-based test bed. Conventional machine learning-based intrusion

detection models were implemented using well-known classification techniques, including extreme learning machine, nearest neighbor, decision-tree, random-forest, support vector machine, naive-bays, and quadratic discriminant analysis. Both deep and conventional machine learning models were evaluated using well-known classification metrics, including receiver operating characteristics, area under curve, precision-recall curve, mean average precision and accuracy of classification. Experimental results of deep IDS models showed promising results for real-world application in anomaly detection systems.

### **2.3.2 Network Intrusion Detection based on Directed Acyclic Graph and Belief Rule based**

Intrusion detection is very important for network situation awareness. While a few methods have been proposed to detect network intrusion, they cannot directly and effectively utilize semi-quantitative information consisting of expert knowledge and quantitative data. Hence, this paper proposes a new detection model based on a directed acyclic graph (DAG) and a belief rule base (BRB). In the proposed model, called DAG-BRB, the DAG is employed to construct a multi-layered BRB model that can avoid explosion of combinations of rule number because of a large number of types of intrusion. To obtain the optimal parameters of the DAG-BRB model, an improved constraint covariance matrix adaption evolution strategy (CMA-ES) is developed that can effectively solve the constraint problem in the BRB. A case study was used to test the efficiency of the proposed DAG-BRB. The results showed that compared with other detection models, the DAG-BRB model has a higher detection rate and can be used in real networks.

## **2.4 Preliminaries**

In this section, we shortly discuss the background information for our study. We start by describing the overview of the IDS/IPS and the SIEM, and introduce the deep learning techniques. Finally, we describe our big data platform for the proposed AI-SIEM system.

### **2.4.1 IDS / IPS and SIEM**

#### **IDS/ IPS**

An intrusion detection system (IDS) monitors the network activity and reports on observation of any security violations [3]. Unlike the IDS, an intrusion prevention system (IPS) can block a detected network connection by closing port or dropping the packets. An IPS has become an indispensable system for most types of organizations or industries owing to the wide growing nature of data and the internet. Nevertheless, intelligent network attacks still persist in today's network, and there are limitations to detect and respond network intrusions by an IPS system [4]. This is because they mainly use less-capable signature-based detection, as opposed to anomaly detection methods. Meanwhile, speedy attacks are occurring more frequently with new intrusion methods [5]. Most of all, the majority of IPS solutions have a high false positive rate and are limited in detecting any unknown or new

attacks. In addition, in [5], the authors presented six limitations for an IPS such as the challenges of volume, accuracy, diversity, dynamics, lowfrequency attacks, and adaptability. These limitations lead to seriously restrict precise decision by an SOC security analyst.

## SIEM

A SIEM has been considered an important component of enterprise networks and security infrastructures, with a focus on enterprise information technology (IT) security, which provides an overall view of the security management. In general, SIEM collects relevant data produced in an organization from various sources, making it possible to detect cyber threats by matching patterns [6]–[7]. The SIEM system allows the consolidation and comprehensive evaluation of security alerts and logs collected from network security systems (e.g., firewall and IDS / IPS). Particularly with analyzing IDS/IPS alerts (security events) in SIEM, the analyst make an effort to find cyber attacks using pre-defined security policies and threshold. Moreover, to discover consolidated malicious behavior, they carry out analyzing correlations between security events and relevant situations based on already known patterns of cyber threats [8]. Security events are continually generated from many types of network security systems (e.g., IPS and FW); thus, they are heterogeneous with an extremely diverse distribution.

This brings challenges to discriminate true positive alerts from false ones in a traditional policy-based threat detection system. Moreover, practice shows that this method of analyzing is extremely complex, high costly and only operable with large personnel effort [9]. For cyber-threat detection, the SIEM analysts spend an immense amount of effort and time to differentiate between true security alerts and false security alerts in collected events. Hence, in recent years, to address this challenge, one of the main focuses within the development of SIEM has been the application of machine-learning and artificial-intelligence (AI)-learning techniques, which is referred to here as AI-based SIEM. Although the application of these techniques has offered improvement in reducing human labor, there are still several challenges for an AI-based SIEM. As mentioned above, there are major limitations such as (1) the comparatively high level of analyst interaction required, (2) lack of labeled data, and (3) constantly evolving attacks [10], [5].

# Chapter 3

## Proposed System and System Architecture

### 3.1 Proposed System

Our proposed system aims at converting a large amount of security events to individual event profiles for processing very large scale data. We developed a generalizable security event analysis method by learning normal and threat patterns from a large amount of collected data, considering the frequency of their occurrence. In this study, we specially propose the method to characterize the data sets using the base points in data preprocessing step. This method can significantly reduce the dimensionality space, which is often the main challenge associated with traditional data mining techniques in log analysis.

- Our event profiling method for applying artificial intelligence techniques, unlike typical sequence-based pattern approaches, provides featured input data to employ various deep-learning techniques. Hence, because our technique is able to facilitate improved classification for true alerts when compared with conventional machine-learning methods, it can remarkably reduce the number of alerts practically provided to the analysts.
- For the applicability, we evaluate our system with real IPS security events from a real security operations center (SOC) and validate its effectiveness through performance metrics, such as the accuracy, true positive rate (TPR), false positive rate (FPR) and the F-measure. Moreover, to evaluate the performance comparison with existing methods, we conducted experiments using the five conventional machine-learning methods (SVM, k- NN, RF, NB and DT). And we also perform an evaluation by applying our method to two benchmark datasets (i.e., NSLKDD, CICIDS2017), which are most commonly used in the field of network intrusion detection research.

#### 3.1.1 Advantages

- For cyber-threat detection, the SIEM analysts spend an immense amount of effort and time to differentiate between true security alerts and false security alerts in collected events.

- **Reduction of False Positives** :The system's focus on discriminating between true positive and false positive alerts helps in reducing the number of false positives, which is a significant issue in traditional intrusion detection systems.
- **Efficient Alert Management** : With fewer false positives, security analysts can prioritize and respond to genuine threats more effectively, improving overall security operations.
- **Real-World Applicability** : Testing with real-world datasets confirms the system's practical utility and effectiveness in actual cybersecurity environments.

## 3.2 System Requirement Analysis

In this we describing concept to detect threats using AI-SIEM (Artificial Intelligence-Security Information and Event Management) technique which is a combination of deep learning algorithms such as FCNN, CNN (Convolution Neural Networks) and LSTM (long short term memory) and this technique works based on events profiling such as attack signatures. Author evaluating propose work performance with conventional algorithms such as SVM, Decision Tree, Random Forest, KNN and Naïve Bayes. Here I am implementing CNN and LSTM algorithms.

### **Propose algorithms consists of following module**

1. **Data Parsing**: This module take input dataset and parse that dataset to create a raw data event model
2. **TF-IDF**: using this module we will convert raw data into event vector which will contains normal and attack signatures
3. **Event Profiling Stage**: Processed data will be splitted into train and test model based on profiling events.

The data sets we are using for testing are large and while building model it's will cause an out of memory error but the kddtrain.csv dataset works perfectly, but to run all algorithms it will take 5 to 10 minutes. You can test remaining datasets also by reducing its size or running it on high configuration system.

### 3.2.1 Feasibility Study

Three key considerations involved in the feasibility analysis are

- **Economical Feasibility**
- **Technical Feasibility**
- **Social Feasibility**

### **Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased

### **Technical Feasibility**

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

### **Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

## **3.3 Data Labeling for Learning**

In this subsection, we discuss the data labeling of security events for supervised learning. As mentioned above, to employ the supervised learning method, a labeled data is essential. For this, analysts should be able to label several months of data heuristically. In other words, analysts need to label the raw events as “Normal” or as “Threat,” based on whether it belongs to a type of attack by analyzing correlations among raw security events. However, owing to a rapidly growing number of security events and unknown cyber threats, the labeling of numerous data is timeconsuming and costly. In addition, it is difficult to acquire the labeled security event dataset based on the action of SOC security experts in the real world. By investigating occurred cyber attacks, most of detected attacks can be categorized as system hacking, denial of service, network attacks, scanning attacks, and suspicious authentication activities. These attack types are determined by the SOC security analysts based on correlation among attack duration time, the number of attacker’s IP, and importance of victim system [10].

## 3.4 Datasets

The two datasets used for testing, are NSLKDD, CICIDS 2017 and which are real datasets collected in the SOC.

### 3.4.1 NSLKDD

The NSLKDD dataset is the new revised version of the KDDCUP99. Tavallaee et al. [11] had discovered a number of duplicated records in the original KDDCUP99 dataset, which had an impact on the performance of model training and evaluation on the dataset. NSLKDD is a refined version of the dataset to address discovered statistical problems. Some advantages over KDDCUP99 are that the complexity can be reduced and bias toward frequent records by machine learning algorithms can be prevented. However, this new version of the dataset still suffers from some of the problems discussed by McHugh [12] and may not be a perfect representation of existing real networks. Because recent NIDS research still uses this dataset for performance evaluations, we believe it is regarded as an effective benchmark to help us compare different methods. The training is performed on KDDTrain data which contain 22 attack types and testing is performed on KDDTest data which contains 17 additional attack types. These attacks can be categorized into four different types with some common properties for training and testing. The four categories of attacks are: Denial of Service (DoS), Probe, Remote to Local (R2L) and User to Root (U2R)! [11].

### 3.4.2 CICIDS 2017

In 2017, the Canadian Institute for Cybersecurity (CIC) published an intrusion detection dataset named CICIDS2017 [13]. This dataset provides the labeled data for the field of network intrusion detection research and contains benign activities and attacks, which was collected for five days log (from Monday to Friday). While the first day log contains normal activity and only includes the benign data, the other days contain the data points for various attacks together with benign data. The number of data points is approximately 2.8 million with 85 features including the label information. The implemented attacks include Brute Force FTP, Brute Force SSH, DoS, Heartbleed, Web Attack, Infiltration, Botnet and DDoS. The dataset has used the B-Profile system [13] to profile the abstract behavior of human interactions and generate naturalistic benign background traffic.

## 3.5 System Architecture

The system architecture of the ThreatSenseIDS Intrusion Detection System (IDS) comprises a complete end-to-end pipeline for cyber-attack detection that employs both classical Machine Learning models as well as advanced Deep Learning architectures. The different components of the system architecture are arranged in layers, where in each layer a task is performed that is very important for the accurate prediction of the threats, the data processing done in an efficient way and the user being able to interact through a graphical interface smoothly. The architecture layers start with the Data Acquisition Layer which is in charge of loading and maintaining the

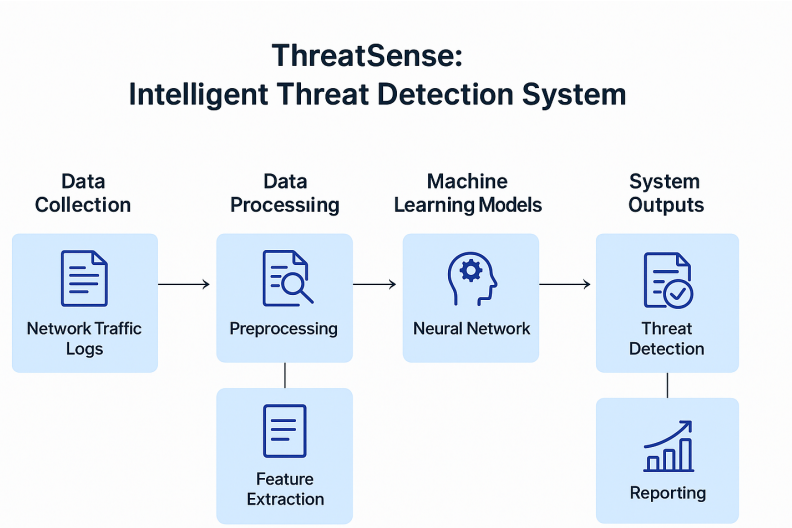


Figure 3.1: System Architecture

KDD Cup dataset that contains network traffic records with features such as duration, protocol type, number of bytes transferred, failed login attempts, connection count, and different flag indicators. This dataset is the base of the whole system since all models get their learning from these features. After the dataset is loaded, it goes to the Data Preprocessing Layer which is the place for doing the crucial changes. This consists of managing the categorical features through Label Encoding or One-Hot Encoding, normalization of the numerical features, and dividing the dataset between the training and testing sets. These preprocessing phases make sure that the models get inputs that are correctly formatted and scaled which will result in more stable and accurate predictions [14].

The data that has been processed is then sent to the Feature Engineering and Selection Layer, where the insignificant or just duplicate features are eliminated and the significant attributes kept. This stage enhances the computational efficiency and also prevents the overfitting of deep learning models like CNN and LSTM, which are very susceptible to this problem. Some of the engineered features such as connection rates, service flags, and error counts among others are very beneficial for the detection of the following: probing, denial-of-service (DoS), remote-to-local (R2L), and user-to-root (U2R) attacks [? ]. Modeling and Training Layer is the backbone of the architecture, which utilizes several algorithms.

Baseline comparison and robustness are ensured by including classical models like KNN (K-Nearest Neighbors), SVM (Support Vector Machine), Random Forest, and Decision Tree as well. Meanwhile, Deep Learning models such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks are the ones that allow the system to acknowledge both the temporal and spatial dependencies in the network traffic. CNN is good at detecting signatures of pattern-based attacks, while LSTM is great at recognizing sequential behavior that is typical in slow-burn attacks. Each model gets trained separately, and the system monitors the accuracy, precision, recall, and F1-score to see how well the model performs [15]. The Evaluation and also Detection Layer takes the output of all the models, compares them and points out the model that has the best prediction performance. It

detects the types of threats that are present in the sample and sorts the behavior as either normal or malicious. The threats that are recognized are DoS attacks, probing, unauthorized access attempts, and privilege escalation attacks. At last, the Graphical User Interface (GUI) Layer that uses Tkinter coordinates the whole process. The user can upload the data sets, see the graphs, and select the algorithms, train the models and get the performance results in a clean and friendly interface immediately. The whole system works like a complete, modular, and scalable IDS solution that is ready for research, presentations, and future deployment.

# Chapter 4

## Models and Testing Types

### 4.1 Model Selection

Model Selection in machine learning is the process of choosing the best suited model for a particular problem. There are two factors to consider before selecting an ML model, the logical reason for selecting a model and the performance of various models. Based on these factors, we can choose the best model for the task at hand and the type of data available [16]. Developing an Intrusion Detection System (IDS) using multiple machine learning models can significantly enhance its accuracy and reliability.

#### 4.1.1 Random Forest

Random Forest creates several decision trees and aggregates the outcomes to get a final forecast. Because each tree learns from a distinct subset of the data, each tree has a slightly different perspective on the issue. Every tree "votes" when it comes time to make a prediction, and the most popular outcome is selected. Random Forest is helpful in threat detection because it can handle big, complicated datasets and find patterns that point to malicious activity. Compared to a single decision tree, it minimizes errors, prevents overfitting, and offers more reliable and accurate threat classification. [6]



Figure 4.1: Random Forest

### 4.1.2 K-Nearest Neighbors (KNN)

K-Nearest Neighbors (KNN) can be utilized in cyber threat detection to find malicious activity. A feature vector is used to represent each event, such as a network request or an attempt to log in. Based on the majority of its neighbors, KNN assesses the "k" nearest past events to determine whether the new event is typical or suspicious. It doesn't require an explicit model because it is a non-parametric, lazy-learning algorithm that readily adjusts to shifting patterns. KNN can become slow and less accurate on large-scale, high-dimensional threat data, despite being straightforward and efficient for small datasets [17].

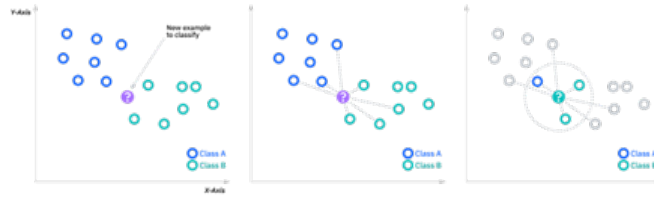


Figure 4.2: K-Nearest Neighbors

### 4.1.3 Support Vector Machine (SVM)

Support Vector Machine is frequently used to categorize occurrences as benign or malevolent. It learns to differentiate between safe activity and threat patterns by analyzing features from network logs, system events, or security datasets. Thanks to a process known as a kernel, which converts the data into a higher-dimensional space where class separation is simpler, SVM's strength is that it performs well even when the data is not perfectly separable.

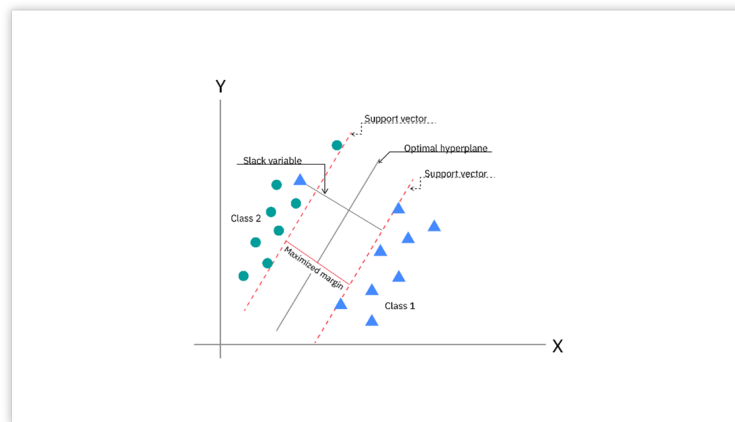


Figure 4.3: Support Vector Machine

#### 4.1.4 Convolutional Neural Network (CNN)

A deep learning model Convolutional Neural Network (CNN) is used to automatically recognize threat patterns in pictures or other visual data. CNNs are used in threat detection systems to identify features in security film or system visual logs, such as shapes, odd items, suspicious activity, or unexpected patterns. A CNN can swiftly determine whether an occurrence is typical or perhaps dangerous by automatically extracting these features, increasing accuracy and minimizing manual analysis. [18]

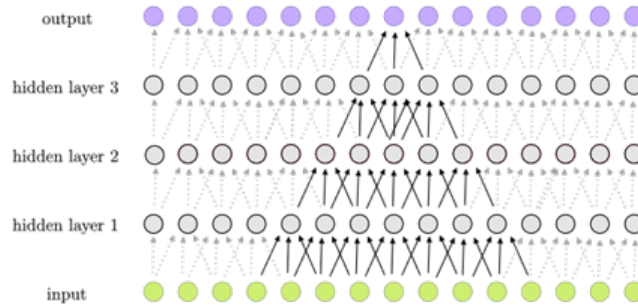


Figure 4.4: Convolutional Neural Network

#### 4.1.5 Decision Tree

A Decision Tree is a supervised machine learning algorithm used for classification and regression. In order to create a tree-like structure where internal nodes reflect feature decisions, branches represent outcomes, and leaf nodes provide the final prediction, it divides data into branches based on feature values. Decision trees can assess security events in threat detection by identifying trends in past data, such as file access, network traffic, and login attempts. By tracking decision routes across the tree, they are able to categorize events as either malicious or normal. Decision trees can overfit if improperly pruned, yet they are quick and simple to understand for small datasets. [19]

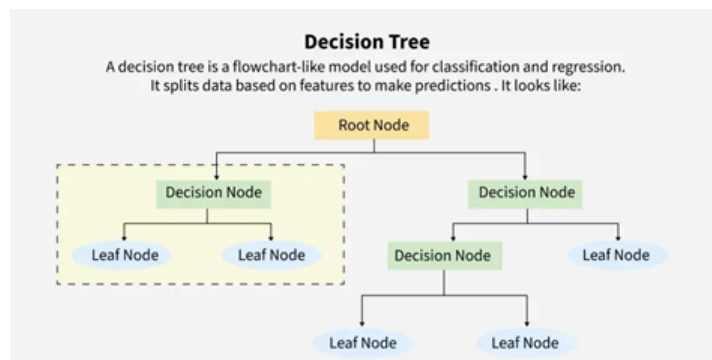


Figure 4.5: Decision Tree

### 4.1.6 Naive Bayes

Naive Bayes is a machine learning classification algorithm that predicts the category of a data point using probability. It assumes that all features are independent of each other. Naive Bayes performs well in many real-world applications such as spam filtering, document categorization and sentiment analysis.

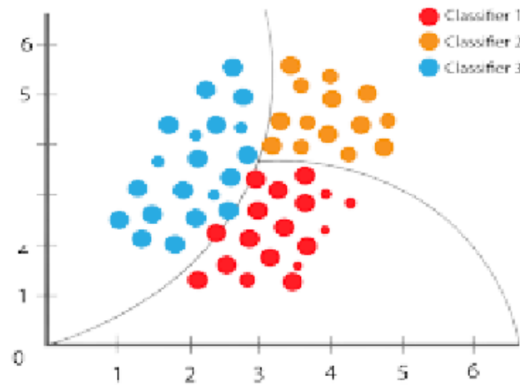


Figure 4.6: Naive Bayes

## 4.2 Testing and Types of Testing being performed

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

### 4.2.1 White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

### 4.2.2 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

### 4.2.3 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### 4.2.4 Test Strategy and Approach

Field testing will be performed manually and functional tests will be written in detail.

#### Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

#### Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

## 4.3 Performance Evaluation Results and Discussion

In the field of machine learning, it is essential to have a clear understanding of how well our model is performing, particularly when it is employed to distinguish between normal network traffic and abnormal traffic. A highly effective tool for evaluating a model's performance is the confusion matrix [20]. It is a tool used to evaluate the performance of a classification model compared to the actual value in the dataset in four parts: True Positives (TP), where the model correctly predicts the positive class. True Negatives (TN), where the model correctly predicts the negative class. False Positives (FP), where the model incorrectly predicts the positive class. False Negatives (FN), where the model incorrectly predicts the negative class. It allow us to calculate how the model is right, how accurate the model predicts the positive class, and how good the model is at detecting the actual positives [21].

Accuracy is the most intuitive performance measure and it is simply a ratio of correctly predicted observations to the total observations. It is a measure of how well the model performs across all classifications.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.1)$$

Precision is the ratio of number of True Positive to the total number of Predicted Positive. It measures, out of the total predicted positive, how many are actually positive.

$$Precision = \frac{TP}{TP + FP} \quad (4.2)$$

Recall is the ratio of number of True Positive to the total number of Actual Positive. It measures, out of the total actual positive, how many are predicted as True Positive.

$$Accuracy = \frac{TP}{TP + FN} \quad (4.3)$$

F-Score is an important evaluation metric for classification that combines Precision & Recall. This is a very useful metric when a dataset has imbalanced classes [22].

$$F-Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (4.4)$$

The performance evaluation metrics of all models in presented in Table. Naive Byes performed the worst for all of the evaluation metrics. The ensemble voting classifier outperformed the individual models, showcasing the advantages of combining multiple classifiers to make predictions. The high accuracy percentage was indicative of the proposed models' effectiveness in successfully distinguishing between attack and non-attack samples. The high precision value suggest that the model has a low false positive rate, minimizing the chances of flagging benign files as attack. The high recall value indicate that the proposed model can successfully identify a large proportion of actual attack flows, reducing the risk of missed detections.

Table 4.1: Performance Evaluation Metrics of All Models

	<b>Accuracy</b>	<b>Precision</b>	<b>Recall</b>	<b>FMeasure</b>
KNN	93.50	91.75	91.70	91.70
Random Forest	91.32	88.12	87.24	87.24
SVM	97.90	94.35	92.74	93.74
Naive Bayes	84.3	82.72	82.43	82.10
Decision Tree	88.42	88.37	88.21	88.21
CNN	98.15	94.39	94.17	94.07

# Chapter 5

## Conclusion and Future Work

### 5.1 Conclusion

In this paper, we have proposed the AI-SIEM system using event profiles and artificial neural networks. The novelty of our work lies in condensing very large-scale data into event profiles and using the deep learning-based detection methods for enhanced cyber-threat detection ability. The AI-SIEM system enables the security analysts to deal with significant security alerts promptly and efficiently by comparing longterm security data. By reducing false positive alerts, it can also help the security analysts to rapidly respond to cyber threats dispersed across a large number of security events.

For the evaluation of performance, we performed a performance comparison using two benchmark datasets (NSLKDD, CICIDS2017) and two datasets collected in the real world. First, based on the comparison experiment with other methods, using widely known benchmark datasets, we showed that our mechanisms can be applied as one of the learning-based models for network intrusion detection. Second, through the evaluation using two real datasets, we presented promising results that our technology also outperformed conventional machine learning methods in terms of accurate classifications.

### 5.2 Future Work

Several potential directions for future work in this study could significantly affect the longevity, and performance of the threat-detection system. Besides the current model that employs TF-IDF-based event profiling to compare different machine-learning and deep-learning algorithms, the project can evolve into a more intelligent, context-aware, and autonomous detection phase.

The first aspect is to incorporate advanced deep-learning architectures into the system such as the Transformer-based models (e.g., BERT, GPT-based anomaly detectors) which are skilled at recognizing the long-term temporal patterns and complex contextual relationships in security event logs. Unlike CNN and LSTM, Transformers can perform their work on the whole sequence at once, which might lead to a significant increase in the detection of slow-evolving, stealthy attack behaviors such as APTs or insider threats. Moreover, one major drawback of current IDS research is the dependence on manual labeling of datasets. Large-scale security logs

labeling is a time-consuming and expensive process hence, future researches can look into semi-supervised and self-supervised learning methods to ease the dependence on labeled data. Techniques such as autoencoders, contrastive learning, and clustering-based anomaly detection can automatically extract patterns from unlabeled raw logs, thus increasing the scalability of the system to real SOC environments.

The third point is that we would like to interlink the real-time data streaming and online learning features, which will let the models for detecting the attacks constantly adapt to the new patterns. This is very important since the attackers usually change their strategies, techniques, and procedures (TTPs). An online learning system would let the knowledge base of the system to be updated instantly rather than waiting for the periodic offline retraining. Moreover, the use of threat intel feeds which comprises IPs and domains that are known to be malicious, MITRE ATT&CK mappings, and behavioral indicators would be a very good idea for the detection part of the system since it will be able to connect the alerts with the already existing global threats and thus, the true-positive rates would go high and the workload for the analysts would reduce considerably.

The last point is that a complete AI-SIEM system with a user-friendly interface, automated alert prioritization, and explainable AI (XAI) components would be readily available for the analyst. Explainability techniques like SHAP or LIME could bring security teams to the level of understanding in the matter of "what" and "why" regarding the model marking a particular event thus making the system more transparent and hence, trustable.

# Contribution

The contributions I have made to the ThreatSense project were substantial and led to its successful delivery. The first step of my involvement was to architect a complex yet comprehensive system, where all the components of data ingestion, model evaluation, etc. were tightly fitted and functioned well together. Exposing the project documentation was another major allocation made on my part. I made sure that the sections contained technical descriptions that were academically acceptable and clear, offering in-depth detail on the system's design, functionality, and evaluation methods. Besides that, I also incorporated the machine learning parts that were necessary for the creation of the main performance metrics like Accuracy, Recall, Precision, and F1-Score.

Along with the above, I also designed and wrote the framework for generating comparative graphical visualizations that made it possible to analyze the performance of different algorithms easily and quickly.

In addition, I was in constant contact with my teammates and participated actively in the designing of a user-friendly graphical interface with Tkinter. One of my contributions was in the designing of the interactive UI elements, the tuning of the user-friendliness, and the incorporation of the real-time updating of the interface. This linking made sure that there was smooth running with no hitches between the backend algorithms and the frontend system, and thus the usability and overall effectiveness of the ThreatSense application were increased.

# Bibliography

- [1] S. S. Roy, A. Mallik, R. Gulati, M. S. Obaidat, and P. V. Krishna, “A deep learning based artificial neural network approach for intrusion detection,” in *International conference on mathematics and computing*. Springer, 2017, pp. 44–53.
- [2] F. R. Alzaabi and A. Mehmood, “A review of recent advances, challenges, and opportunities in malicious insider threat detection using machine learning methods,” *IEEE Access*, vol. 12, pp. 30 907–30 927, 2024.
- [3] N. Hubballi and V. Suryanarayanan, “False alarm minimization techniques in signature-based intrusion detection systems: A survey,” *Computer Communications*, vol. 49, pp. 1–17, 2014.
- [4] P. POORNACHANDRAN, A. AL-NEMRAT, and S. VENKATRAMAN, “Deep learning approach for intelligent intrusion detection system,” *IEEE Access*, 2024.
- [5] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, “A deep learning approach to network intrusion detection,” *IEEE transactions on emerging topics in computational intelligence*, vol. 2, no. 1, pp. 41–50, 2018.
- [6] T.-F. Yen, A. Oprea, K. Onarlioglu, T. Leetham, W. Robertson, A. Juels, and E. Kirda, “Beehive: Large-scale log analysis for detecting suspicious activity in enterprise networks,” in *Proceedings of the 29th annual computer security applications conference*, 2013, pp. 199–208.
- [7] J. Lee, J. Kim, I. Kim, and K. Han, “Cyber threat detection based on artificial neural networks using event profiles,” *Ieee Access*, vol. 7, pp. 165 607–165 626, 2019.
- [8] S. S. Sekharan and K. Kandasamy, “Profiling siem tools and correlation engines for security analytics,” in *2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*. IEEE, 2017, pp. 717–721.
- [9] K.-O. Detken, T. Rix, C. Kleiner, B. Hellmann, and L. Renners, “Siem approach for a higher level of it security in enterprise networks,” in *2015 IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, vol. 1. IEEE, 2015, pp. 322–327.

- [10] K. Veeramachaneni, I. Arnaldo, V. Korrapati, C. Bassias, and K. Li, “Ai<sup>2</sup>: training a big data machine to defend,” in *2016 IEEE 2nd international conference on big data security on cloud (BigDataSecurity), IEEE international conference on high performance and smart computing (HPSC), and IEEE international conference on intelligent data and security (IDS)*. IEEE, 2016, pp. 49–54.
- [11] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, “A detailed analysis of the kdd cup 99 data set,” in *2009 IEEE symposium on computational intelligence for security and defense applications*. Ieee, 2009, pp. 1–6.
- [12] J. McHugh, “Testing intrusion detection systems: a critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory,” *ACM Transactions on Information and System Security (TISSEC)*, vol. 3, no. 4, pp. 262–294, 2000.
- [13] I. Sharafaldin, A. H. Lashkari, A. A. Ghorbani *et al.*, “Toward generating a new intrusion detection dataset and intrusion traffic characterization.” *ICISSp*, vol. 1, no. 2018, pp. 108–116, 2018.
- [14] S. Naseer, Y. Saleem, S. Khalid, M. K. Bashir, J. Han, M. M. Iqbal, and K. Han, “Enhanced network anomaly detection based on deep neural networks,” *IEEE access*, vol. 6, pp. 48 231–48 246, 2018.
- [15] B.-C. Zhang, G.-Y. Hu, Z.-J. Zhou, Y.-M. Zhang, P.-L. Qiao, and L.-L. Chang, “Network intrusion detection based on directed acyclic graph and belief rule base,” *Etri Journal*, vol. 39, no. 4, pp. 592–604, 2017.
- [16] D. Yuan, J. Huang, X. Yang, and J. Cui, “Improved random forest classification approach based on hybrid clustering selection,” in *2020 Chinese Automation Congress (CAC)*. IEEE, 2020, pp. 1559–1563.
- [17] Y. Liao and V. R. Vemuri, “Use of k-nearest neighbor classifier for intrusion detection,” *Computers & security*, vol. 21, no. 5, pp. 439–448, 2002.
- [18] J. Wu, “Introduction to convolutional neural networks,” *National Key Lab for Novel Software Technology. Nanjing University. China*, vol. 5, no. 23, p. 495, 2017.
- [19] K. A. Taher, B. M. Y. Jisan, and M. M. Rahman, “Network intrusion detection using supervised machine learning technique with feature selection,” in *2019 International conference on robotics, electrical and signal processing techniques (ICREST)*. IEEE, 2019, pp. 643–646.
- [20] Y. Wang, “Teaching performance evaluation model based on machine learning,” in *2022 14th International Conference on Measuring Technology and Mechatronics Automation (ICMTMA)*. IEEE, 2022, pp. 843–847.
- [21] G. Alkhatib, “Exploring machine learning hypothesis testing,” *International Journal of Computers*, vol. 9, 2024.
- [22] A. Bhandari, “Understanding & interpreting confusion matrix in machine learning,” *Analytics Vidhya*, 2023.