INTRUSION DETECTION SYSTEMS FOR SQL DATABASES USING MACHINE LEARNING

A Thesis

Submitted towards the Requirements for the Award of Degree of

Doctor of Philosophy

In

Department of Computer Science & Engineering

Under the Faculty of Engineering & Technology

By

CHADCHANKAR AMARNATH SHIVANAND

(Enrollment No. 161588517473)

Under the Supervision of

Dr. Balveer Singh

Professor



2026

P.K. UNIVERSITY

Shivpuri, M.P. India-473665

P.K.UNIVERSITY
SHIVPURI (M.P.)

University Established Under section 2F of UGC ACT 1956 Vide MP Government Act No17 of 2015

CERTIFICATE OF THE SUPERVISOR

This is to certify that the work entitled **"INTRUSION DETECTION SYSTEMS FOR SQL DATABASES USING MACHINE LEARNING"** is a piece of research Work done by **CHADCHANKAR AMARNATH SHIVANAND** Under My Guidance and Supervision for the degree of Doctor of Philosophy of University (M.P) India.

I certify that the candidate has put an attendance of more than 240 day

with me. To the best of my knowledge and belief the thesis:

I –Embodies the work of the candidate himself /herself.

II –Has duly been completed.

III –Fulfil the requirement of the ordinance relating to the Ph.D. degree of the University.

Date:…….. **Signature of the Supervisor**



P.K.UNIVERSITY
SHIVPURI (M.P.)

University Established Under section 2F of UGC ACT 1956 Vide MP Government Act No17 of 2015

DECLARATION BY THE CANDIDATE

I declare that the thesis entitled" **INTRUSION DETECTION SYSTEMS FOR SQL DATABASES USING MACHINE LEARNING"** is my own work conducted under the supervision of **Dr. BALVEER SINGH,** Department of COMPUTER SCIENCE & ENGINEERING, Faculty of ENGINEERING & TECHNOLOGY, at P.K. University, Shivpuri, (M.P.)

Approved by Research Degree Committee. I have put more than 240 days of attendance with supervisor at the center.

I further declare that to the best of my knowledge the thesis does not contain my part of any work has been submitted for the award of any degree either in this University or in any other University Without proper citation.

Date:-…….. **Signature of the Candidate**

Place: P.K. University, Shivpuri, (M.P.)

P.K.UNIVERSITY
SHIVPURI (M.P.)

University Established Undersection 2F of UGC ACT 1956 Vide MP Government Act No17 of 2015

COPY RIGHT TRANSFER CERTIFICATE

Title of the Thesis**: "INTRUSION DETECTION SYSTEMS FOR SQL DATABASES USING MACHINE LEARNING"**

Candidate's Name: CHADCHANKAR AMARNATH

SHIVANAND

COPY RIGHT TRANSFER

The undersigned hereby assigns to the P.K. University, Shivpuri (M.P.) all copyrights that exist in and for the above thesis submitted for the award of the Ph.D. Degree.

Date: **CHADCHANKAR AMARNATH SHIVANAND**

Note: However, the author may reproduce/publish or authorize others to reproduce, material extracted verbatim from the thesis or derivative of the thesis for author's personal use provided that the source and the University's copyright notice are indicated.

P.K.UNIVERSITY
SHIVPURI (M.P.)

University Established Undersection 2F of UGC ACT 1956 Vide MP Government Act No17 of 2015

FORWARDING LETTER OF HEAD OF INSTITUTION

The Ph.D. thesis entitled **"INTRUSION DETECTION SYSTEMS FOR SQL DATABASES USING MACHINE LEARNING"** Submitted by **CHADCHANKAR AMARNATH SHIVANAND, Enrollment No.: - 161588517473,** university in six copies. The candidate has paid the necessary fees and there are No-dues outstanding against her.

Name: -

Seal…….

Date……

Place: - P.K. University, Shivpuri, (M.P.)

(Signature of Head of Institution
where the Candidate was
registered for Ph.D. degree)

Signature of the Supervisor

Date: …….

Place: - P.K. University, Shivpuri, (M.P.)

P.K.UNIVERSITY
SHIVPURI (M.P.)

University Established Under section 2f of UGC ACT 1956 Vide MP Government Act No 17 of 2015

Ref. No.:- Ph.D/PKU/2026/04/52 Date:..18/04/26...

Pre – Ph.D. Submission Presentation Certificate

This is to certify that **Mr. Chadchankar Amarnath Shivanand,** Enrollment No: **161588517473,** research scholar under the supervision of **Prof. Balveer Singh**, Department of **Computer Science & Engineering,** Faculty of **Engineering & Technology** has given his open Pre-Ph.D. submission presentation on the approved topic **"INTRUSION DETECTION SYSTEMS FOR SQL DATA BASES USING MACHINE LEARNING."** on Date **18/04/2026** at Time **11:00** AM in the P.K. University, Shivpuri (MP).

Dean Academic
P.K. University
Shivpuri (M.P.)

ADDRESS: VILL: THANRA, TEHSIL: KARERA, NH-27, DIST: SHIVPURI, M.P. 473665,
MOB: 7241115088, Email:registrar.pkuniversity@gmail.com

P.K.University
Shivpuri (M.P.)

Enrolment Number 161588517473

Select Course: Ph.D. in Computer Science & Engg. (Course Work)

Select Semester: 1

Result

Enrolment Number : 161588517473

Candidate Name : CHADACHANKAR AMARNATH SHIVANAND

Course : Ph.D. in Computer Science & Engg. (Course Work)

Father's Name : SHIVANAND SHARNAPPA CHADACHANKAR

Year/Sem : 1

Mother's Name : MANGAL

Session : 2021-22

Subject Name	Internal T	Internal T	Internal P	Internal P	External T	External T	External P	External P	
Research Methodology	N/A	N/A	N/A	N/A	50	37	N/A	N/A	37 / 50
Subject Specialization	N/A	N/A	N/A	N/A	100	75	N/A	N/A	75 / 100
Review of Literature	N/A	N/A	100	72	N/A	N/A	N/A	N/A	72 / 100

Marks Obtained 184 Result Pass Max Marks 250

- Student must pass in Theory and Practical separately.
- For pass the candidate is required to obtain 40% marks in each paper and 50% marks in aggregate.
- For pass the Ph.D candidate is required to obtain 65% marks in aggregate.

ACKNOWLEDGEMENT

The Acknowledgement of a research endeavor is an opportunity to express appreciation for the joint effort of those who have been sources of motivation and support, and who have contributed invaluable knowledge towards achieving success in the pursuit of knowledge. With utmost humility and pride, I reflect on the inspiring individuals who have propelled my work forward, and I am filled with immense pleasure, gratitude and thanks for their unwavering dedication.

First for most I would like to thank almighty god for perseverance, courage and strength bestowed on me to undertake this project.

For most, I extend my deepest appreciation to my esteemed Supervisor, **Prof. Dr. Balveer Singh, Prof. Dr. Monika Tripathi** Department of Computer Science & Engineering**,** Faculty of Engineering & Technology, P.K. University, Shivpuri, (M.P.) without whom this work would not have come to fruition. His exceptional guidance, unwavering support kind cooperation and provision of a conductive work environmental have been invaluable beyond measure. I remain indebted to her instilling in me a relentless pursuit of excellence, a strong sense of honesty, and respect for ethical principles that govern our profession.

Furthermore, I extend my sincere thanks to the **Mr. Jagdeesh Prasad Sharma** Chancellor, P.K University Shivpuri (M.P) for providing the necessary infrastructure and facilities that were essential during my research work. My profound sense of gratitude, faith, and we also thanks to **Prof. (Dr.) Mukesh Chansoriya** Vice - Chancellor, P.K University Shivpuri (M.P) for his generous and valuable support throughout my research.

I take this pleasant opportunity to express my deep sense of gratitude and reverences to his holiness **Dr. Jitendra Kumar Mishra** Director; P.K University Shivpuri (M.P) for his constant encouragement and providing me with the required facilities that enabled me to complete my project work.

Dr. Jitendra Malik Registrar, P.K University, Shivpuri (M.P.), who has inspired me with his encouraging guidance and support right from the beginning of this work in spite of his busy schedule. I am short of words to thank him for his affectionate behavior and patience throughout the duration of research work.

I express my gratitude to **Dr. Aiman Fatima,** Dean Academics, of P.K. University, Shivpuri, (M.P.)

I would also like to extend my gratitude to and deep appreciation to respected **Dr. Ashish Vishwakarma,** Dean Research of P.K. University, (M.P.)

I owe a great deal of application and gratitude to entire faculty members of Faculty of Engineering and Technology P.K. University, **Dr. Manish Kumar, In-charge Ph.D. cell**, **Dr. Monika Tripathi** and all non-teaching staff of Department of Computer science and Engineering without their support it would have been difficult to shape my research work.

I also express my gratitude to **Miss. Nisha Yadav,** Librarian of P.K University along with **Mr. Anand Kumar**, Assistant Librarian and other staff of P.K University. I am also thankful to **Mr. Pankaj Singh, Mr. Alok Khare**, and **Mr. Prushottam Yadav** I.T. Cell of P.K. University.

I have no words to express my gratitude to my beloved parent's dear brother's for their love, support and constant encouragement during the entire course of my research work.

Last but not least I would like to honestly appreciate the fact that it is my pleasure to have an opportunity to pursue my higher education at P.K. University, Shivpuri (M.P). I owe my permanent debt towards it; this is having the special place in my heart for shaping my care.

I must also express my love and affection to my family members for his understanding and unwavering love during my research tenure, Lastly, I am highly grateful to God, the almighty, for giving me courage and strength to face the most difficult times of life and for answering my prayers in a special way. I remain grateful for all that I have achieved in my life thus far.

Date:

Place: **(CHADCHANKAR AMARNATH SHIVANAND)**

TABLE OF CONTENTS

Sr. No.	PARTICULARS		PAGE.NO.
1	**CHAPTER – 1 INTRODUCTION**		1-30
	1.1	Overview	1-2
	1.2	Intrusion Detection Systems	2-8
	1.3	Machine Learning	8-22
	1.4	Intrusion Detection System Using Machine Learning	22-26
	1.5	Ids State-Of-The-Art Works	26-29
	1.6	Objectives of the study	30-30
2	**CHAPTER – 2 REVIEW OF LITERATURE**		31-77
3	**CHAPTER – 3 RESEARCH METHODOLOGY**		78-106
	3.1	Architecture	78-80
	3.2	Data Preprocessing	80-81
	3.3	Web Application	81-82
	3.4	Attacks	82-82
	3.5	Machine Learning	83-84
	3.6	SCADA Dataset	84-91
	3.7	KDD Cup’99	91-93
	3.8	Siamese Network Usage Scenarios Overview	93-101
	3.9	Outlier-Based Zero-Day Attacks Detection	101-106
4	**CHAPTER-4 EXPERIMENT AND RESULTS**		107-150
	4.1	Overview	107-111
	4.2	SCADA (Supervisory Control and Data Acquisition)	111-121
	4.3	MQTT IDS Dataset Generation	121-126
	4.4	MQTT (Message Queuing Telemetry Transport)	126-131
	4.5	SCADA Dataset Results	131-134
	4.6	CICIDS2017 Dataset Results	135-136
	4.7	KDD Cup’99 and NSL-KDD Dataset Results	136-145
	4.8	Autoencoder Results	145-150
5	**CHAPTER-5 LIMITATIONS OF STUDY**		151-151
6	**CHAPTER-6 CONCLUSION**		152-155
7	**CHAPTER-7 FUTURE SCOPE**		156-156
8	**REFERENCES**		157-173
9	**LIST OF PUBLICATIONS**		

List of Figures

Figure No	Figure Name	Page No
Figure 1.1	Neural network components	12
Figure 1.2	How neurons operate	13
Figure 1.3	LSTM gates	14
Figure 1.4	LSTM forget gate	15
Figure 1.5	LSTM update gate/input gate	16
Figure 1.6	LSTM output gate	16
Figure 1.7	The architecture of CNN	17
Figure 1.8	Decision tree	18
Figure 1.9	Random Forest	19
Figure 1.10	Intrusion detection systems	23
Figure 3.1	System Process	78
Figure 3.2	Network Architecture	79
Figure 3.3	Data Preprocessing	80
Figure 3.4	SCADA System Architecture	85
Figure 3.5	SCADA Network High-Level Architecture	85
Figure 3.6	SCADA: Preprocessing Stages	89
Figure 3.7	SCADA: The trends of the Rate of Change of Register4 readings for different scenarios	89
Figure 3.8	Siamese Network Usage Scenarios Overview	94
Figure 3.8	Siamese Network for Intrusion Detection (Classification)	94
Figure 3.9	Siamese Network Loss Curve (Non-converging case)	98
Figure 3.10	Siamese Network Loss Curve (Converging case) - 1	98
Figure 3.11	Siamese Network Loss Curve (Converging case) – 2	98
Figure 3.12	Autoencoder Convergence Curve	104
Figure 4.1	Classification Accuracy	109
Figure 4.2	Modeling Time	109
Figure 4.3	Testing Time	110
Figure 4.3	SCADA: Anomaly Detection Overall Accuracy (5-fold cross-validation)	112
Figure 4.4	SCADA: Affected Component Overall Classification Accuracy (5-fold cross-validation)	114
Figure 4.5	SCADA: Scenarios Overall Classification Accuracy, Single Scenario (5-fold cross-validation)	118
Figure 4.6	SCADA: Scenarios Overall Classification Accuracy, Two Probable Scenarios (5-fold cross-validation)	119

Figure 4.7	SCADA: Scenarios Overall Accuracy Classification, One or Two Scenario(s) Based on 75% and 85% Confidence Intervals	121
Figure 4.8	MQTT-IoT-IDS2020: Network Architecture	123
Figure 4.9	MQTT-IoT-IDS2020: Overall Detection Accuracy Trend using Different ML Techniques	127
Figure 4.10	MQTT-IoT-IDS2020: Benign Class Performance	130
Figure 4.11	MQTT-IoT-IDS2020: MQTT BF Class Performance	131
Figure 4.12	MQTT-IoT-IDS2020: Weighted Average	131
Figure 4.13	SCADA Dataset k-NN (k=30) and Siamese Network (SN) (30	134
Figure 4.14	CICIDS2017 Autoencoder Detection Results Summary Per	143
Figure 4.15	CICIDS2017 Autoencoder and One-Class SVM Comparison	145
Figure 4.16	KDD Cup'99 Autoencoder and One-Class SVM Comparison	149
Figure 4.17	NSL-KDD Autoencoder and One-Class SVM Comparison	150

List of Tables

Table 1.1: Supervised learning vs unsupervised learning 11
Table 1.2: A description of the KDD Cup 99 dataset 20
Table 1.3: Details of NSL-KDD dataset 22
Table 3.1 SCADA: PLC Registers Extracted Bits Representation 86
Table 3.2 SCADA: Dataset Scenarios,Operational Scenarios, Affected Components . .88
Table 3.3 KDD Cup'99 Classes and Corresponding Number of Instances 92
Table 3.4 NSL-KDD Classes and Corresponding Number of Instances 92
Table 3.5 CICIDS Classes and Corresponding Number of Instances 93
Table 3.6 CICIDS2017 Attacks……102
Table 4.1: Results With 20000 Records 107
Table 4.2: F-score for 20000 Record Data 108
Table 4.3SCADA Results:Experiment1-Anomaly Detection(5-fold cross-validation)113
Table 4.4 SCADA Results: Experiment 2 - Affected Component Classification (5-fold cross-validation) 115
Table 4.5 SCADA: Distribution of Probabilistic Classification of Scenarios……..118
Table 4.6 SCADA: Co-relation of scenarios that are misclassified based on one probable scenario and correctly reported with the second probable one (Calculated based on k-NN experiment) 120
Table 4.7 MQTT-IoT-IDS2020: Feature List and Description…….125
Table 4.8 MQTT-IoT-IDS2020: Instances Distribution 126
Table 4.9 MQTT-IoT-IDS2020: Overall Detection Accuracy 127
Table 4.10 MQTT-IoT-IDS2020 Results: LR - k-NN - DT (5-fold cross-validation)..128
Table 4.11 MQTT-IoT-IDS2020 Results: DT-RF-SVM-NB (5-fold cross-validation)129
Table 4.12 Sample Confusion Matrix 1322
Table 4.13 Siamese Network: SCADA Classification Confusion Matrix (j = 5)…….133
Table 4.14 Siamese Network: SCADA Classification Accuracy Using Different j Votes…133
Table 4.15 Siamese Network: CICIDS2017 Classification Accuracy Using Different j Votes 1355

Table 4.16 Siamese Network: CICIDS2017 Classification Accuracy Using Different j Votes…135
Table 4.17 Siamese Network: KDD Cup'99 Classification Confusion Matrix (j = 5) 136
Table 4.18 Siamese Network: KDD Cup'99 Classification Accuracy Using Different j Votes 137
Table 4.19 Siamese Network: NSL-KDD Classification Confusion Matrix (j = 5) 137
Table 4.20 Siamese Network: NSL-KDD Classification Accuracy Using Different j Votes 138
Table 4.21 Recent IDS Studies for Multi-Class Classification Performance..... 140
Table 4.22 Zero-Day Detection: CICIDS2017 Autoencoder Results 142
Table 4.23 Zero-Day Detection: CICIDS2017 One-Class SVM Results 144
Table 4.24 Zero-Day Detection: KDD Cup'99 Autoencoder Results..... 146
Table 4.25 Zero-Day Detection: NSL-KDD Autoencoder Results..... 147
Table 4.26 Zero-Day Detection: NSL-KDD Performance Comparison 148
Table 4.27 Zero-Day Detection: KDD Cup'99 One-Class SVM Results..... 148
Table 4.28 Zero-Day Detection: NSL-KDD One-Class SVM Results..... 149

ABSTRACT

SQL Injection is still among the worst security flaws, exposing users' private data and causing financial losses. The most current OWASP Top 10 study ranks injection attacks as the top vulnerability, and the frequency of these attacks is on the rise. By definition, Intrusion Detection System (IDS) rules that rely on static signatures are a common component of traditional defensive systems. These rules are great for preventing known attacks but aren't effective against zero-day threats. Many recent studies have used machine learning methods, which may identify previously unseen threats but can be performance-heavy depending on the methodology. To add insult to injury, some intrusion detection systems scour database server logs for information, while others focus on gathering traffic entering the web app across the network or the web app host. A web application host, a MySQL database server, and a Datiphy appliance node placed between the two are the two sources of traffic that will be collected in this project. Through our examination of these two datasets as well as an additional dataset that is correlated with them, we have proven that the accuracy achieved with the correlated dataset using algorithms like decision trees and rule based approaches is comparable to that of a neural network algorithm, but with substantially better performance. Newly interwoven technology and diverse systems have emerged as a result of the digital transformation of both general- and special-purpose networks., as our dependence on networked technologies has grown. Cyberattacks have grown in both frequency and sophistication in recent years, in part because of the dynamic given the current state of technology and the exponential growth of connected devices. The capacity to analyses network traffic using Intrusion Detection Systems (IDS) is a crucial component of any networking security toolset. Using ML and DL techniques, researchers have created intrusion detection systems (IDS) to cope with new and zero-day cyberattacks, which are becoming more common and sophisticated. But developing IDS is difficult since there aren't enough big, realistic, and current datasets.

This thesis suggests studying In both general-purpose and special-purpose networks, ML and DL algorithms can identify both known and unknown attacks using empirical means. This thesis delves deeper into the topic of how DL and ML algorithms may learn with less input while maintaining a high level of accuracy. Specifically, six ML approaches are tested on an IoT dataset designed for specific purposes. Anomaly detection in special-purpose networks is examined, along with its limits and difficulties.

Newly interwoven technology and diverse systems have emerged as a result of the digital transformation of both general- and special-purpose networks., as our dependence on networked technologies has grown. The proliferation and sophistication of cyberattacks in recent years are at least partially attributable to the dynamic nature of the technology environment, where the number of networked devices is always growing. An important part of any networking security toolbox allows one to examine data sent by a network by use of intrusion detection systems. Using ML and DL techniques, researchers have created intrusion detection systems (IDS) to deal with new and zero-day cyber threats, which are becoming more common and sophisticated. But developing IDS is difficult since there aren't enough big, realistic, and current datasets. This thesis suggests studying in both general-purpose and special-purpose networks, ML and DL algorithms can identify both known and unknown attacks. using empirical means. This thesis delves deeper into the topic of how DL and ML algorithms may learn with less input while maintaining a high level of accuracy. Specifically, six ML approaches are tested on an IoT dataset designed for specific purposes. Anomaly detection in special-purpose networks is examined, along with its limits and difficulties.

Keywords: Intrusion Detection System (IDS), SQL Injection Attack, Machine Learning, Network security

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

SQL Injection and other web attacks are old, still it continues to jeopardize the privacy of individuals and the security of a company and government suffer loss of money. This is particularly true when it is taking into account that not old attack vectors are continuously emerging and that old ones are continuously developing. Industry and security businesses heavily fund web attack mitigation; however, most mitigation solutions are restricted and are actively being explored to be broken by current research.

Signature detection is considered one of the typical methods of traditional online attack mitigation, and it involves the use of the static analysis of the incoming online traffic. Development of a signature that is unique to cyberattacks is what this strategy is all about; once a firewall or other security appliance identifies this signature, potentially block the suspect traffic. Despite being fast and potentially applicable in real-time to defend network resources, this technique has a limitation: it can only detect familiar attacks.

Another method of preventing SQL injection attack on the web is by analysing the syntax of the SQL queries coming in; an attack like that will be detected when a query is realised to be faulty. However, this method necessitates a lot of knowledge of the program, however, it can perform a terrific task of identifying new threats with false queries and has high detection scores in general.

A method to identify SQL injections that is currently under investigation use techniques based on machine learning. It is typical in this area of research to employ the decision trees, SVMs, NN, rule-based learning techniques. The capability to detect new attacks is one of the strengths of these techniques. The potential drawback of these strategies is that, with the help of the algorithm applied, the processing time could potentially increase.

It is crucial to have access to credible data in studies examining the above SQL detection strategies and other ones. Several current studies made use of online application and server logs, or web traffic which is captured as it is received by the application. The information

we are going to use in our proposed approach is retrieved in a database server of a Datiphy appliance and the data is retrieved in the incoming and outgoing traffic of the web application on a network node. We build our first two datasets with the data collected on traffic at the two locations and by combining the events in the former two datasets we have our third dataset.

In any scenario where it is necessary to process vast amount of data in a short time, it is challenging to enhance the efficiency and speed of a classification system, and in intrusion detection issue domain, it is even more so. In this study, we are examining algorithms that are faster and more precise than others, including neural networks, although they might be less efficient during the process. In that regard, we constantly seek new ways of enhancing the speed, efficiency, accuracy, and novelty of machine learning in signature detection.

It is also well recognized that ongoing research on intrusion detection systems (IDS) has the challenge of collecting realistic attack data; several studies rely on data from 1999, which is seen to be far from typical of contemporary web-based assaults. Instead of using current traffic, many researchers choose to generate their own, with automated creation tools like SQL map being among the most used. Whether standalone or integrated into a bigger project like Metasploit, this is a tool that genuine attackers and security researchers are using. We did a lot of testing with sqlmap traffic generation for our project, and we discovered that the data was easy to categorize since it consistently had similar characteristics. Our strategy uses machine learning trained on actual traffic generated automatically using SQL injection attacks that were manually developed.

1.2 INTRUSION DETECTION SYSTEMS

Among the primary roles of an IDS is maintaining a watch on any suspicious or unlawful activity within a network or within the system of the network. Typically, a SIEM system will notify the administrator or consolidate all of the relevant information at a single point of an incursion or breach, in the event that a security incident occurs. Using alert filtering algorithms and combining the information of numerous sources, a SIEM system can identify the difference between legal and dangerous activity.

The numerous spheres of influence that IDSs are between individual PCs and large networks. Nowadays, the two primary classes to describe such systems are NIDSs and HIDSs. HIDS would be a system that monitors critical files in the OS, but NIDS would be one that monitors all outgoing network traffic. The other method of categorizing IDS is based on detection method. Signiture-based detection and anomaly-based detection are two methods of detecting harmful patterns and abnormal traffic, respectively, two of the most popular variations. The former uses machine learning to discover signatures and the latter uses history to detect deviations of a model of good traffic. Another commonly used variant is reputation-based detection, which is the recognition of potential threats, which are rated in relation to their reputation. Reactive capabilities to identified intrusion are part of a number of IDS solutions. Systems capable of responding to possible intrusion are frequently called intrusion prevention systems (IPS). IDS can be more helpful with the assistance of specialized tools; a honeypot can be employed to detect and identify malicious traffic.

Comparison with firewalls

Despite their shared interest in keeping networks safe, Firewalls and the developed Intrusion Detection Systems (IDS) to accomplish distinct goals. Traditional firewalls, in contrast to next-generation firewalls, rely on a static set of rules to decide which Connections to allow or reject. Assuming the criteria are suitable, it prohibits incursions implicitly. To put it simply, firewalls block unauthorized access from outside networks while remaining silent in the face of an assault from inside. Following the occurrence of a suspected intrusion, an IDS will provide a description of it and sound an alert. Internal system attacks are other types of threat that an IDS monitors. Traditional procedure for doing this includes monitoring network traffic for typical computer assault heuristics and patterns (also called signatures), and then taking measures to notify operators. Intrusion prevention systems, which act similarly to application layer firewalls in controlling access, terminate connections.

Intrusion detection category

Network or host detection location or signature or anomaly-based detection technique each provide distinct methods in which IDSs are classified.

Analyzed activity

Network intrusion detection systems

Network intrusion detection systems (NIDS) are put in place to monitor all network traffic, whether it's incoming or leaving. The program examines all data packets traveling across the network that compares them to an inventory of previously identified dangers. A warning may be issued to the administrator whenever an assault is detected or strange activity is recognized. Protecting networks from unauthorized access is the main objective of Network bases Intrusion detection system any device or the whole network.

Host intrusion detection systems

Each host or device on a network runs its own host intrusion detection system (HIDS). When a HIDS detects unusual behavior, it will notify the user or administrator. The system solely analyzes packets coming into and going out of the device. It compares the current state of the system files to an earlier snapshot. For the administrator to investigate, a notification is provided in the event that important system files are altered or removed. Critical infrastructure computers, which are not subject to routine configuration changes, are a good example of a system that uses HIDS.

One way to identify any efforts at firewall penetration is to install NIDS on the subnet, which includes firewalls. Although it would be great to monitor inbound and outbound data, doing so might cause the network to slow down and cause a bottleneck. It is usual practice to model NIDS using OPNET and NetSim. Additionally, NID systems may analyze signatures for comparable packets to identify potentially hazardous ones and discard them if they match the entries in the NIDS. Online NIDS, also called inline NIDS, and off-line NIDS, sometimes called tap mode, are the main two tpes of NIDS designs when we look at them through the lens of the system interaction property. Real-time network intrusion detection systems operate online. It determines whether an Ethernet packet constitutes an

attack by analyzing them and applying certain standards. The reason of offline NIDS is to identify attacks based on stored data evaluates it via many steps.

Combining NIDS with additional technologies may further improve detection and prediction rates. The complicated structure of Artificial Neural Networks (ANNs) makes it time-consuming to analyze large amounts of data, yet ANN-based IDSs can handle it because of its hidden layers and non-linear modeling. This makes it easier for intrusion detection systems to identify patterns about incursion. Security solutions that detect intrusions using artificial neural networks aid in the creation of a two-tiered early warning system by learning from past errors in attack prediction. Being able to accept single values in upper layer, the outcome of first layer as input in the second layer allows the system to automatically recognize new, unexpected patterns in the network. This system can averagely withstand 99.9 percent of network assaults, including DOS, Probe, Remote-to-Local, and user-to-root attacks detection and classification rate.

Detection method

Signature-based

Network traffic analysis is the key to threat detection in signature-based intrusion detection systems for certain patterns, such as known malicious instruction sequences or bytes. This word "signature" comes from the world of antivirus software, where it describes certain patterns that have been recognized. Despite signature-based IDS's ease of use in detecting known attacks, it struggles to identify novel assaults for which no pattern exists.

Anomaly-based

Because malware is always changing, anomaly-based intrusion detection systems were created to find new types of threats. Making a model of reliable behavior using machine learning and then comparing it to fresh behavior is the main idea. Compared to classic signature-based intrusion detection systems, The ability to tailor model training to specific applications and hardware configurations gives machine learning-based solutions a leg up when it comes to general characteristics. This method can find attacks that didn't exist before, but it might also have false positives, whereby harmless but previously unseen behavior is mistakenly labelled as harmful. The detection method is time-consuming and

negatively impacts the performance of most current IDSs. The detection process's classification accuracy is enhanced by an efficient feature selection approach.

Analytics for User and Entity Behavior (UEBA) and Network Traffic Analysis (NTA) are two new kinds of systems that Gartner is taking a look at; UEBA is a development of the user behavior analytics category. When it comes to compromised user machines or accounts, NTA specializes in dealing with both malevolent insiders and targeted external assaults. Some companies have chosen NTA over more conventional IDS, according to Gartner.

Intrusion prevention

While it is possible for certain systems to try to prevent intrusion attempts, it is not anticipated or required of a monitoring system to do so. Attempts to detect, record, and report intrusions the main roles that IDPS play in preventing and detecting intrusions. Further uses for IDPS include helping businesses find policy violations, keeping track of current threats, and discouraging employees from breaking company policy. These days, almost any company's security architecture wouldn't be complete without intrusion detection systems.

IDPS usually keep track of data pertaining to events that have been seen, alert security managers to significant occurrences that have been noticed, and generate reports. In addition to detecting threats, many IDPS can take action to stop them in their tracks. One of their many reaction strategies is for the IDPS to halt the assault automatically. Another is to alter the nature of the attack or the security environment (for as by resetting a firewall).

Network security appliances that keep an eye out for any suspicious behavior in a network or system are called computer networks that are designed to detect and prevent intrusions. Identifying harmful behavior, logging information about it, reporting it, and attempting to block or halt it are the key duties of an intrusion prevention system.

Because they both keep an eye out for suspicious behavior in system operations and network traffic, intrusion prevention systems are seen as an extension of intrusion detection systems. The most notable distinction is that intrusion prevention systems may actively stop or block detected intrusions, in contrast to intrusion detection systems, and they are installed in-line. Depending on the severity of the threat, an IPS may notify the user, discard

harmful may be used to reset the connection, block traffic from the problematic IP address, or even reset packets. Moreover, an IPS may defragment streams of packets, repair errors in cyclic redundancy checks (CRCs), lessen problems with TCP sequencing, and eliminate needless decisions made at the transport and network layers.

Classification

Intrusion prevention systems may be broadly classified into four types:

Network-based intrusion prevention system (NIPS): investigates protocol operations to detect malicious traffic throughout the whole network.

Wireless intrusion prevention system (WIPS): examine protocols for wireless networking with the goal of keep an eye on a network for any suspicious activity.

Network behavior analysis (NBA): analyses data sent by a network in order to spot inappropriate behavior, violations of rules, and distributed denial of service (DDoS) attacks, among other potential dangers.

Host-based intrusion prevention system (HIPS): installed software that analyzes activities happening inside a single host and monitors it for suspicious behavior.

Detection methods

Three main detection approaches are used by most intrusion prevention systems: signature-based, statistical anomaly-based, and stateful protocol analysis.

Signature-based detection: digital signature Network packets are monitored by Intrusion Detection Systems (IDS), which compare them against signatures., which are pre-defined patterns of attacks. Although it's the most effective and easiest way, it won't catch new assaults or variations on old ones.

Statistical anomaly-based detection: A system that detects intrusions by looking for unusual patterns will keep an eye on all network traffic and compare it to a predetermined standard. By establishing a standard, the baseline will reveal the protocols and bandwidth types that are considered "normal" for the network in question. But if the baselines aren't smartly set up, it can trigger a false positive for valid bandwidth use. Using Matthews'

correlation coefficient, ensemble models have achieved a 99.73% success rate in detecting malicious network traffic.

Stateful protocol analysis detection: Using "pre-determined profiles of generally accepted definitions of benign activity" as a comparison point, this technique finds protocol state deviations. Although it can learn and follow protocol states, it uses a lot of power.

Limitations

Intrusion detection systems have noise as one of its main adversaries. The incidence of false alarms could be caused by faulty packets that are the result of software defects, DNS data that is incorrect, or local packets that have escaped.

Many times, the number of false alerts is far higher than the number of real attacks. Reason being, the frequency of false alarms greatly outnumbers the number of actual assaults, the former is often disregarded.

Unfortunately, many of these assaults target out-of-date software versions. In order to reduce risks, a signature library that is updated on a regular basis is necessary. Using an out-of-date signature database might leave the IDS open to newer attacks.

In the future, there could be a lag time for signature-based intrusion detection systems when new threats are discovered before the signature is deployed. Because of this delay, the intrusion detection system will miss the danger.

Weak identification and authentication techniques or protocols in the network cannot be remedied by it. Security systems are unable to thwart malicious activity when an attacker obtains access because of inadequate authentication procedures.

Even when they are encrypted, most intrusion detection systems still do not process them. Hence, an undetected intrusion into the network might take place thanks to the encrypted packet, which can wait for more serious incursions to happen.

Information pertaining to intrusion detection software is derived from a unique identifier for every IP packet that arrives to a network. Assuming the IP packet's declared network address is correct, this is useful. On the other hand, the IP packet's address can be garbled or spoofed.

Network intrusion detection systems (NIDS) are often vulnerable to protocol-based assaults, much like network hosts, because of their very design and the need to analyze recorded protocols in real-time. Malicious data poses a threat to network intrusion detection systems (NIDS) or assaults on the TCP/IP stack.

When it comes to cloud computing security, users' privacy demands might vary widely. Whether they're dealing with businesses or regular people, they provide the same level of protection.

1.3 MACHINE LEARNING

1.3.1 Machine learning tasks

Creating an algorithm to machine learning can be divided into two main stages. The various models are trained in different ways. Widely used frameworks in this regard and to construct machine learning models are scikit-learn, Tensor flow, PyTorch, Matlab and Weka. The framework reigns when it comes to optimization of the method or the amount of parameters available.

Intrusion detection is particularly popular in three applications of machine learning models, namely, classification, regression, and reconstruction. Classification sorts entries into numerous categories, and may include the categories of normal or attack or even specific families of attacks. To locate continuous values, so is the chance that a specified input is an assault, regression (also known as "prediction") is employed. Finally, there is only a type of neural networks capable of reconstruction. To obtain representation learning enables the network to learn the properties, this task tries to deflate and reflate the input data in a bid to reconstruct it.

Machine learning methods can be trained either supervised or unsupervised. One such model is that of the neural networks which can be taught in either direction. Most models are learned through supervised learning, where the inputs and correct results are provided through the dataset. These outputs are assigned to their corresponding inputs and this is a mathematical function that the algorithm learns to model. Two traditional supervised training problems are classification and regression. On the contrary, unsupervised training

does not utilize findings of the training dataset. It aims to get acquainted with interesting data structures. One of event which will not require supervision is reconstruction.

The machine learning models would require testing after they have been trained so as to measure their effectiveness. This assessment can not be done on data that was not part of the training set. Had the model been exposed to this information previously, or the correct results in the case of supervised learning, then the evaluation would have been biased. A validation set can also be used to compare the values of alternative parameters (such as a learning rate or the number of neurons. Obtaining the value which did the next best performer on the validation set step following training. We then subject the entire network to a test on the test data. Moreover, a validation set needs fresh data; usually, a subset that is set aside with this purpose in mind.

1.3.2 Unsupervised learning vs Supervised learning

The process of training a computer with the use of tagged datasets is known as supervised learning. What this means, literally, is that part of the marked data has the right answer stamped on it. Training on labelled data allows one to anticipate unanticipated outcomes; this is the essence of supervised learning. For example, if one want to teach a computer to estimate the actual travel time between two locations, then they will need to collect and evaluate certain data. The weather, holidays, time of day, and selected route are all examples of the kinds of data that may fall into this category. All of these specifics are inputs. It seems to reason that getting there would take more time when it's pouring, but a machine requires numbers. So, precise data like length of time required to go from a starting point and relevant variables like time, weather condition, route, etc., are needed to construct a dataset that can be trained. The computer will be able to forecast trip times by analyzing the provided data and identifying relationships between various pieces of information. The need to monitor a machine learning model is eliminated with unsupervised learning. Unsupervised learning, on the other hand, works with datasets that are not labelled, since the model will attempt to find the information on its own. Unsupervised learning allows machines to discover any kind of data pattern and also aids in determining which characteristics are necessary for data classification.

Table 1.1: Supervised learning vs unsupervised learning

	Supervised learning	Unsupervised learning
Process	Input and output data are given.	Only input data is given.
Input data	The machine is trained using labeled data.	The machine is not given unlabeled data.
Algorithms used	SVM, NN, Random Forest, Linear and Logistics regression, Classification trees.	Different categoriez: K-means, Cluster algorithms, Hierarchical clustering, and so on.
Computational complexity	Simple.	Complex.
Use of data	Uses training data and relate input and output results.	Does not use output data.
Accuracy of results	Accurate and trustworthy.	Less accurate and trustworthy.
Real time learning	Learning is offline.	Real-time.
Number of classes	Known.	Unknown.
Main drawbacks	Big data is a challenge.	No precise information in regards to data sorting, and the output is not known.

Semi-supervised learning

Another alternative to unsupervised learning is semi-supervised learning. This approach combines a combination of supervised and unsupervised learning. A small amount of labelled data is mixed with a large quantity of unlabeled data during training.

1.3.2 Neural Networks (NN)

To put it simply, in order to process data, a Neural Network uses an interconnected network of neurons. Figure 1.1 shows that a Neural Network primarily consists of an input layer, two or more hidden layers, and an output layer. Neurons, connections, weights, a propagation function, and a learning algorithm comprise each of these layers.

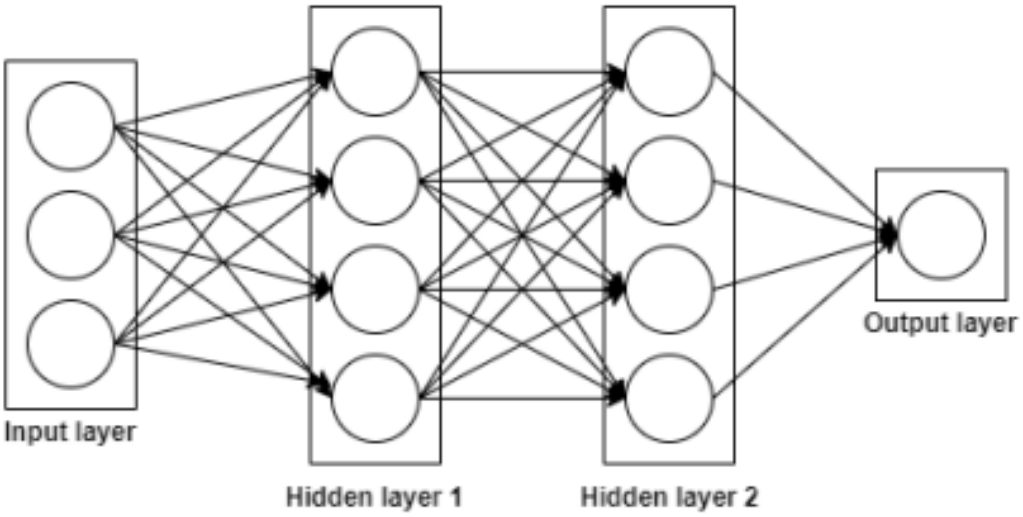


Figure 1.1: Neural network components

Machine learning systems, sometimes known as constructed using neurons, sometimes called nodes. A Neural Network's neurons take data into the network and send it out again. This signifies that a Neural network is represented by the input and output neurons. The neurons that make up the input layer can only take out data; they cannot take in any more information. Neurones in the output layer, on the other hand, take in data but do not produce any outputs of their own. An integral aspect of every Neural Network are its connections and weights. As an example, the output is sent from hidden layer 1 to hidden layer 2 via the connectors, where the neurons in that layer receive it as input. The significance of the input is simply represented by the weight of each of these links. To calculate the inputs to a neuron, one uses a propagation function. When training the data, this function is usually used.

When deciding on or changing the connection weights, the learning rule function is crucial. It facilitates the Neural Network's production of the intended results. During training, these functions are used in the same way as propagation functions.

According to [42], a Neural Network runs in the following way: At first, the input layer is used to transmit data into the network., which is where each of the circles stands for a feature. Second, the data processing concealed layers are called layers. How many levels there are may be adjusted to suit the data processing requirements. Simply said, the more hidden layers there are, the better the accuracy of the findings. Neurons in one layer feed data into the next via the nodes in the layer below. When information is received, it is given

more weight and a bias is imposed to it. Layers that provide output of a neural network take the data processed in the network's last hidden layer and turn it into the final product. Where w is the weight, x is the input vector, and b is the neuron bias, a mathematical representation of NN is illustrated in Figure 1.2 and equation 1.1, $\odot$ is neuronal output, f is the activation function, and y is the element-wise multiplication.

$$y(x) = f(w \odot x + b)$$

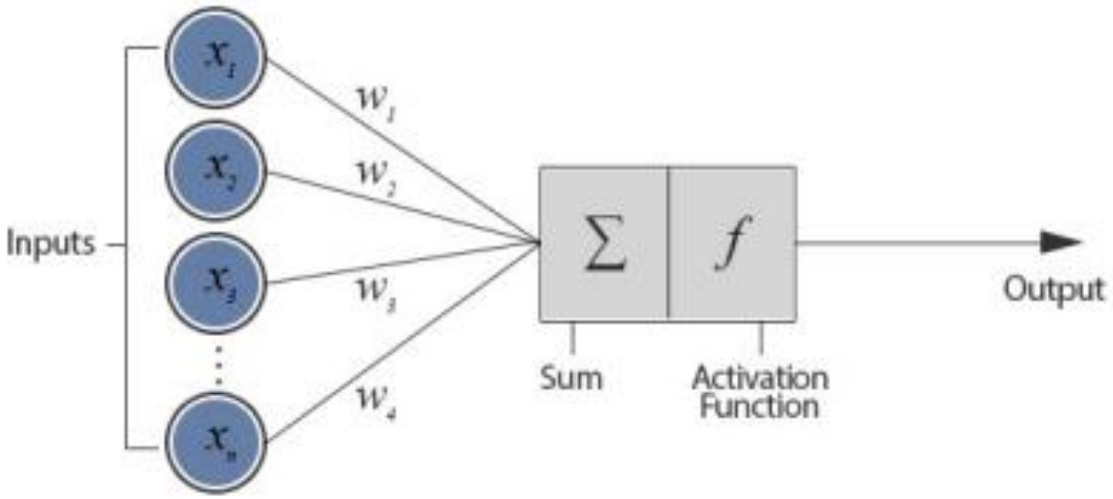


Figure 1.2: How neurons operate

1.3.3 Long Short-Term Memory (LSTM)

Given that LSTM is a subset of RNNs, we should begin by defining Recurrent Neural Networks (RNNs). Through the use of sequential information to one's advantage is the fundamental premise of RNNs. Conventional Neural Networks make the mistake of assuming that inputs and outputs are completely separate in many real-world situations. The rationale for this is because knowing the prior value is crucial for predicting the current one. Because it can execute calculations using the data stored in its "memory," which includes details about what has been considered in the computation so far, the RNN the term used to describe this kind of network.

Learning from long-term dependencies is a capability of LSTM RNNs. [21]. This algorithm can retain information for extended periods of time, just as its name suggests (Long Short-

Term Memory). Its three main procedures—an input gate, an output gate, and a forget gate—make it work. A thorough explanation of LSTM is given in Figure 1.3.

The forget gate determines the amount of information about the past that is remembered. Estimating the probability by means of a sigmoid or squashing function, which limits the output to a 0–1 range, it decides what information should be removed from the cell at a given timestamp. It takes the input xt and the state ht-1 into consideration, as shown in Figure 1.3, and then outputs a number indicating whether to retain or discard the data.

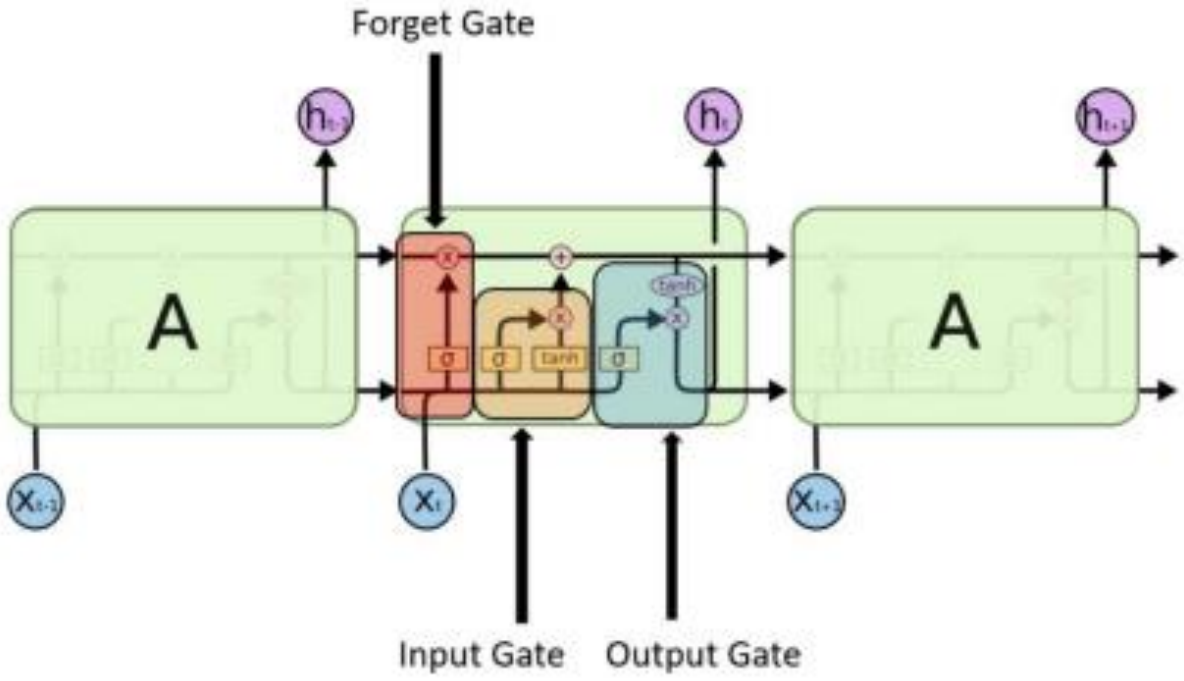


Figure 1.3: LSTM gates

using a value between 0 (delete) and 1 (keep) for each cell state value Ct-1. As seen in 1.8, the forget gate equation.

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

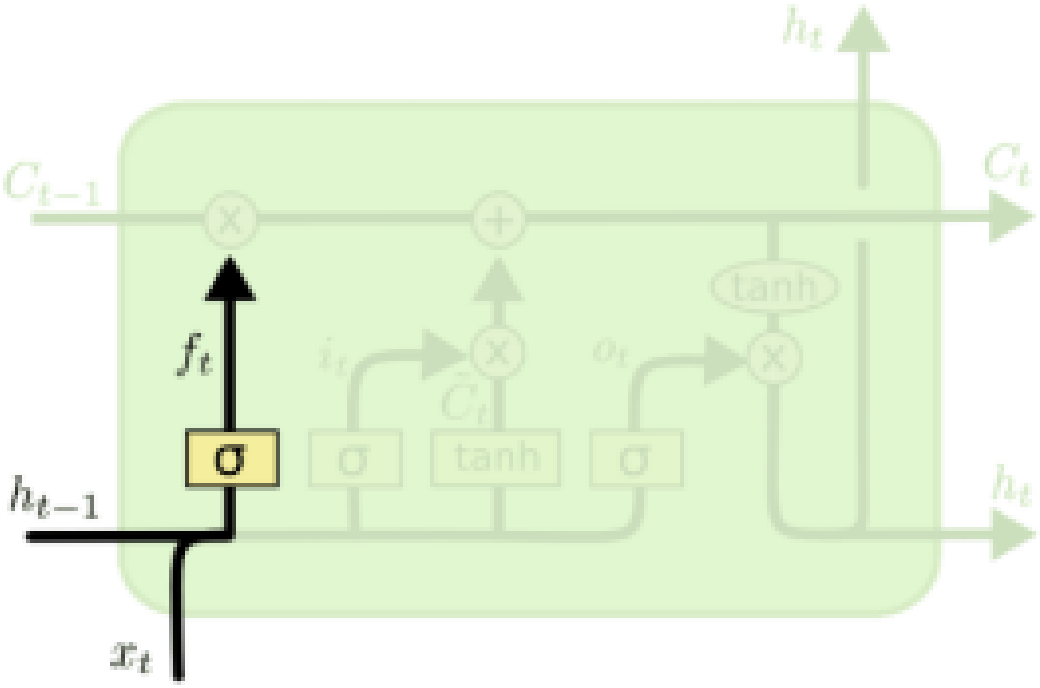


Figure 1.4: LSTM forget gate

• The update/input gate ascertains the quantity of data to be added to the current state. The sigmoid function determines which values pass through and which do not in this gate, similar to the forget gate in equation 1.9. More specifically, a value between -1 and 1 is assigned to each of the provided variables to indicate their relative significance using a tanh function, as seen in equation 1.4. LSTM input/update gate is seen in Figure 1.5.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = tanh(W_X \cdot [h_{t-1}, x_t] + b_C)$$

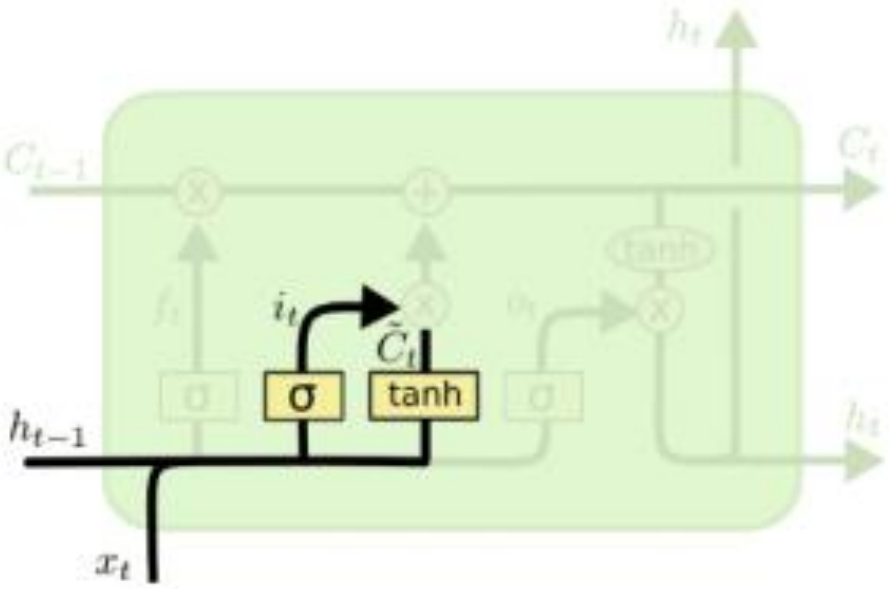


Figure 1.5: LSTM update gate/input gate

The last gate that determines which data point is going to be sent to the output, which can be seen in 1.5. The forget and tanh functions serve the same purpose as they do in input gates and sigmoid functions. In 1.5 and 1.6, we can see the two equations, respectively.

$$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$$

$$h_t = o_t * tanh(C_t)$$

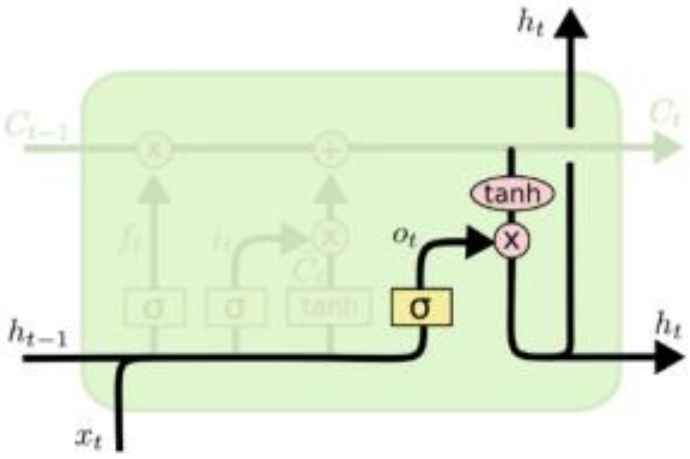


Figure 1.6: LSTM output gate

1.3.4 Convolutional Neural Network (CNN)

Consisting of neurons that include weights and biases, CNN [31] is similar to Neural Networks. Inputs and outputs are two aspects of a neuron. An activation function takes in weights that indicate the data's relevance as inputs and returns an output based on the results.

In contrast to NN, which takes a vector as input, CNNs get a multi-channelled picture as input, allowing them to operate across volumes. This is the fundamental distinction. The three main layers of a convolutional neural network (CNN) are input, convolutional, and output, as shown in Figure 1.7. On the other hand, CNN can handle text too;

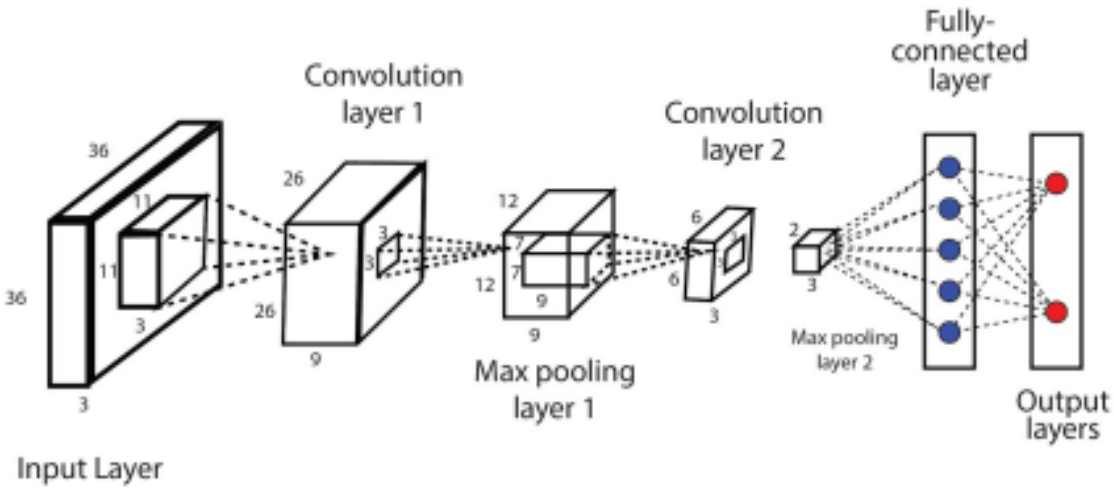


Figure 1.7: The architecture of CNN

Convolutional neural networks (CNNs) have input layers that are linked directly to convolutional layers, which are the fundamental components of CNNs. Extraction and learning of picture information for use in detection methods is the main function of convolutional layers. Pixel values in the input layer have both height and weight; filters employ these values in order to generate outputs that provide characteristics with less dimensions by circling the input layer.

Padding is a CNN primary function. To process an image, the kernel adds a certain amount of pixels, called padding. What makes it work is that it takes a picture and adds pixels to its boundary. The convolutional neural network (CNN) kernel traverses the picture, examines each pixel, after which it adjusts the resulting data by scaling it up or down.. The kernel is

helped in processing the scanned picture by this procedure, and the extra data added to the image frame by the padding allows the kernel to cover more area. Visual analysis becomes more precise when images are padded.

One way to boost a CNN's processing capacity is to decrease the number of parameters. Two methods that do Two examples of this are max-pooling and average-pooling. When compared to average pooling, the mean value from the filter, max-pooling, as its name suggests, takes the greatest value. As a result, pooling is used for dimensionality reduction, whereas padding is employed sparingly.

Random Forest (RF)

When it comes to supervised machine learning, RF is a method that relies heavily on categorization to address problems. In this method, the output accuracy is inversely proportional to the number of Decision Trees. After receiving data and training them to make predictions, the Random Forest algorithm will use a vote system to determine which forecast is the best option. A decision tree is shown in Figure 1.8.

Picking random dataset samples is the first stage of the Random Forest algorithm. The next stage is to build decision trees for all of the chosen samples and then compile their prediction outcomes. After the data is collected, a vote procedure is carried out to choose the most accurate forecast as the ultimate answer. Figure 1.9 provides a basic example of the functionality.

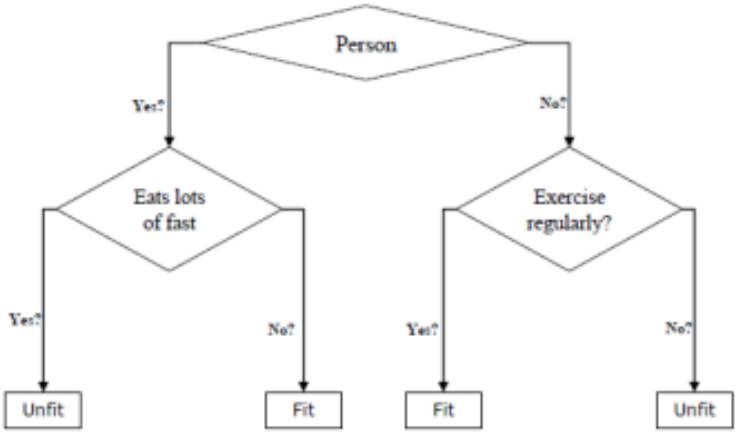


Figure 1.8: Decision tree

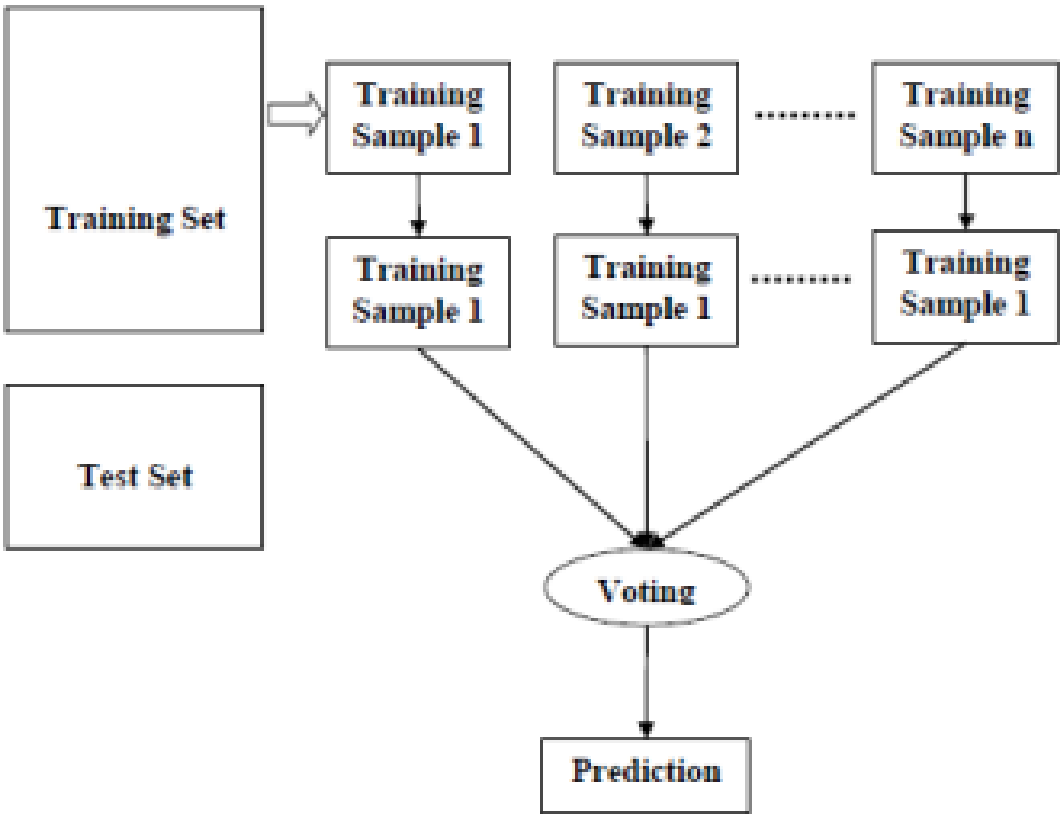


Figure 1.9: Random Forest

The three parts that make up a decision tree [64] are the nodes, which stand for features or attributes, the connections, which represent the rules or choices, and the leaves, which reflect the results.

1.3.5 A comparison between NSL-KDD and KDD Cup 99 (KDD99)

KDD Cup 99

Researchers know and utilize this dataset the best. It is used by the researchers to test out machine learning methods for anomaly identification. The traffic included in this dataset—a subset of the DARPA dataset from 1998—was collected utilizing a virtual environment. An important part of for testing purposes, there is the KDD test dataset, which is part of the KDD Cup 99 benchmark dataset. Classifiers are trained using the remaining 90% of the dataset and the other 10% comprises various attacks and normal connections. There are 4,898,431 entries in the collection that pertain to a single connection; each record has 41

characteristics that are either classified as assaults or considered regular traffic. There are four groups into which these characteristics fall:

• **Basic features:** are read directly extracting information such as protocol type, service, flag, and bytes from the packet header without first examining its contents.

• **Content features:** the analysis and determination of the contents of TCP packets.

• **Time features:** gives details on how long it takes for data to travel from an converting an IP address from its initial location to its final destination.

• **Traffic features:** upon a multi-connected window they rely. It works well for extended assaults.

Another thing about KDD Cup 99 is that it has four primary types of attacks:

• **Probe:** conducted attacks in order to gather system information in order to identify known vulnerabilities. In order to launch further attacks, data collected on such vulnerabilities is used.

• **DoS:** A denial-of-service attack is one that blocks legitimate users from reaching the system.

• **U2R:** A user may get administrative access by launching a "User to Root" attack, which took advantage of security holes in the system. As an example, the hacker may get access to a user account and thereafter look for methods to elevate his privileges.

• **R2L:** In order to access a distant system without a user account, root-to-local assaults are used.

Table 1.2: A description of the KDD Cup 99 dataset

	Whole dataset		10% training set		test set	
Attack category	**Number of instances**	**(%)**	**Number of instances**	**(%)**	**Number of instances**	**(%)**
Normal	492 798	19.86%	97 278	19.69%	60 593	19.48%
Probe	41 102	0.84%	4 107	0.83%	4 166	1.34%
DoS	3 883 370	79.30%	391 458	79.24%	229 853	73.94%
U2R	52	0.00%	52	0.01%	70	0.02%
R2L	1 126	0.02%	1 126	0.23%	16 347	5.26%

This dataset is heavily criticized for a number of reasons:

• Mathematical tasks are difficult.

• There is a great deal of intricacy in both the training and testing sets.

• Machine learning algorithms are affected by the redundancy.

• There is an increase in attack traffic compared to regular traffic.

• There is no practical link between attack categories.

• Attack dispersion detection accuracy is poor.

NSL-KDD

An improved dataset, NSL-KDD, was introduced in 2009 to address the issues of the KDD Cup 99, which includes duplicate records (75% and 78%) and redundant records (78%). By picking a fair number of features from NSL-KDD eliminated problems such as duplicated entries in the training dataset and duplicate records in the testing dataset on the KDD99 dataset. Given that there is a complete absence of duplicate records, it will not be biased while training the machine, which helps to improve the classifier's performance. Instead of picking a small subset at random, it would be more efficient to test the whole dataset, as both the training and testing sets include fewer numbers of occurrences.

There is a total of 37 attacks in the NSL-KDD dataset. Out of them, 21 are found on the test dataset, and every single one of them is in the training dataset as well. Probe, Denial of Service, U2R, and R2L are the four main categories of assaults in NSL-KDD which are identical to those in the KDD99 dataset.

There is a total of 125,973 occurrences in the training dataset, with 67,343 representing typical traffic. Out of a total of 22,850 occurrences, 9,711 represent regular traffic in the test dataset.

Table 1.3: Details of NSL-KDD dataset

Dataset type	Number of records					
	Records	Normal	DoS	Probe	U2R	L2R
NSL-KDD Train+ (full NSL-KDD train set)	125 973	67 343	45 927	11 656	52	995
NSL-KDD Train+ 20% (A 20% subset of the KDD Train+)	25 192	13 449	9 234	2 289	11	209
NSL-KDD Test+ (The full NSL-KDD test set)	22 544	9 711	7 458	2 421	200	2 754
NSL-KDD Test-21 (A subset of the KDDTest+)	11 850	2 152	4 342	2 402	200	2754

1.4 INTRUSION DETECTION SYSTEM USING MACHINE LEARNING

There has been continuous R&D on intrusion detection systems since 1987. An automated monitoring system's value in protecting information technology increased recently as a result of the proliferation of risks linked to computers and networks. One function of monitoring systems or networks is intrusion detection for signs of intrusions. Intruders try to breach security measures or access a computer or network without authorisation, which might threaten its availability, confidentiality, or integrity. They result from unauthorized users wanting to get more unlawful access, authorized people misusing their authority, and hackers breaking into systems over the Internet.

To identify intrusions, two primary methods are employed: abuse detection and anomaly detection. The ability to communicate any matches with them while they are engaged in the current activity is made possible by the pre-recording representation of specific patterns for intrusions, which also acts as the basis for misuse detection. Signatures are patterns of known attack vectors. After that, a classifier trained using network traffic information may differentiate between the two groups. Several data mining approaches, such as association criteria and frequent episodes, were used to extract the discriminative features, which are composed of various statistics related to network traffic. However, when dealing with new types and patterns of attacks, it may be very difficult and time-consuming to collect data samples that are correctly labelled. Anomaly detection, however, labels as an invasion any network activity that substantially differs from typical patterns, after training models for typical traffic patterns. Although neural networks are widely acknowledged as a useful tool for adaptive pattern recognition, their use is severely constrained by their processing overhead and training cycles. This is especially true in intrusion detection situations where a large quantity of relevant data is required.

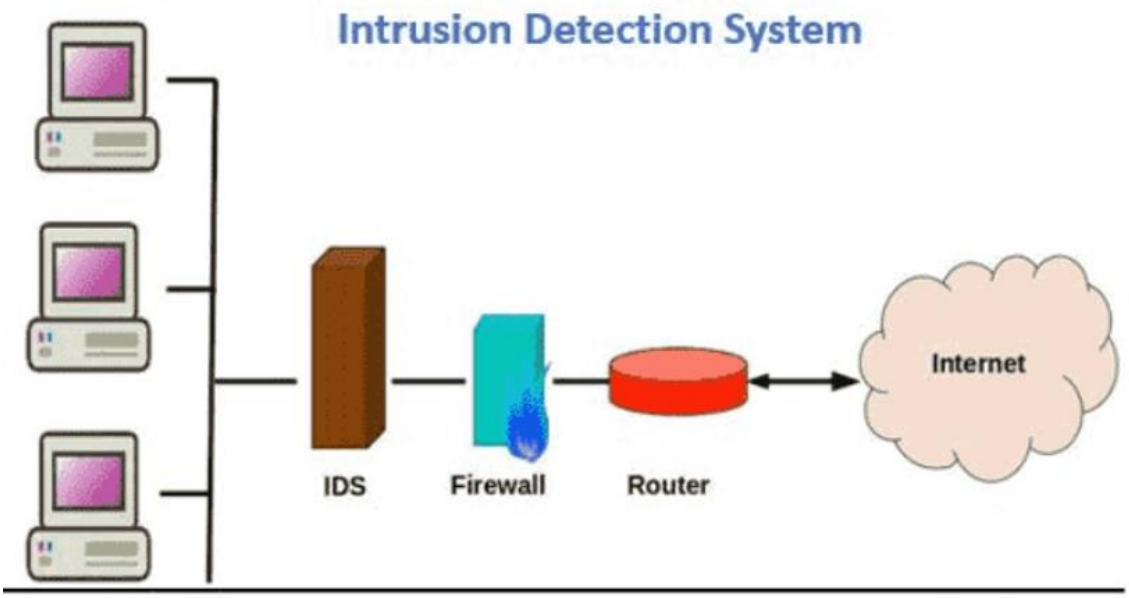


Figure 1.10 Intrusion detection systems

One kind of cyberattack that has persisted for quite some time is SQL Injection. Attackers may get unauthorized access to a database using this method, which entails contaminating application input fields with harmful SQL code. Serious repercussions, including compromised systems and data breaches, may result from this. Cybersecurity is one of the areas that has seen a rise in interest in AI and ML in recent years. An concept to improve security and combat advanced attack methods was to use AI to forecast SQL Injection attempts. An AI model that can assess application input data may be trained to spot patterns that suggest a SQL Injection attack is present.

Conventional security measures for SQL Injection attacks are either static pattern matching or dependent on oversimplified rule-based techniques. But well-planned assaults may sometimes evade these techniques. At this point, it is crucial to have AI that can forecast SQL Injection assaults. Due to the fact that cybercriminals are always improving their techniques, it is difficult to depend just on conventional methods of prediction, which is why AI-based prediction is necessary. Systems driven by AI are better able to detect SQL Injection attempts because they are capable of processing large datasets, finding previously unseen patterns, and adjusting to new attack strategies.

Predictions powered by AI are significant because they can improve detection accuracy. By analyzing attack data from the past, AI models can spot trends, no matter how little, that

might otherwise go undetected by more conventional means. They can lessen the duration and quantity of false positives that genuine user activities are interrupted if they do this. Furthermore, AI may be used as a preventative measure by constantly watching and safeguarding applications from any dangers, such as new and previously unknown SQL Injection attacks.

Thanks to its capacity to sift through mountains of data, identify trends, and provide predictions, AI, and ML in particular, has shown promising results in several cybersecurity domains. Security experts may improve their capacity to identify and counter SQL injection threats by using AI.

Advantages of Using AI to Forecast SQL Injection Attacks

— Extreme Precision: The number of false positives and negatives may be reduced by AI models are able to achieve remarkable precision when it comes to differentiating between valid and malicious SQL queries.

— Real-time Detection: Predictions powered by AI can swiftly evaluate incoming inquiries, allowing for a prompt reaction to real-time threats.

— Adaptability: The model is more resistant to new SQL injection tactics because it can adjust to different attack patterns.

— Reduced Manual Effort: Detection automation frees free allow security personnel to focus on other critical tasks, making better use of available resources.

— Enhanced Security: By using AI-based prediction, online apps may greatly enhance their security measures, protecting critical data and thwarting illegal access.

Developing an Intrusion Detection System (IDS) For Network Security Using Machine Learning

In order to link and share interconnecting various devices and systems over the Internet, the "things" that comprise the IoT are equipped with sensors, software, and other technologies. All sorts of things, from simple household appliances to intricate factory machinery, belong under this group. Technology pertaining to the Internet of Things (IoT) in commercial settings is fundamental to the IIoT, which allows for the attainment of very

high levels of performance, productivity, and efficiency. The manufacturing processes rely on it heavily for controlling machine operations, managing supply chains, and doing predictive maintenance. By enabling remote patient monitoring, the Internet of Medical Things (IoMT) is transforming healthcare and is a medical use case of the Internet of Things (IoT). Because of this, doctors may take immediate action, which enhances the efficacy of clinical treatments. Health care devices that make use of the Internet of Things include fitness trackers and high-tech devices like cuffs that measure heart rate.

For each of the Internet of Things' three main building blocks is crucial to the smooth functioning of IoT systems. The sensors and actuators that make up the Layer of Perception, also known as the Physical Layer —are responsible for gathering data about the surrounding environment and interacting with the actual world. The next layer, the Network Layer, acts as a communication backbone by linking all of the sensors and actuators to the internet using wireless protocols including Wi-Fi, Bluetooth, and Long Term Evolution (LTE). The Application Layer allows users to make decisions using real-time data from the Internet of Things, which leverages processed data to provide tailored services. This layer enables a variety of applications, from medical diagnostics to corporate analytics.

It is critical to implement effective IDS in order to protect IoT networks from the aforementioned security risks. Intrusion detection systems may be broadly categorized into three primary types: HIDS, which are installed on individual devices and monitor system logs and incoming and outgoing traffic for signs of malicious activity; NIDS, which detect suspicious behavior across the entire network; as well as Hybrid IDS, which provide enhanced protection by integrating elements of both NIDS and HIDs. The multi-layered nature of this security solution makes it better able to detect and mitigate risks on the network and individual devices, making IoT systems more resilient to cyberattacks. By enabling systems to learn from data patterns, machine learning improves IDS's anomaly detection capabilities. The primary component here that aids in determining which data attributes are crucial for threat detection is feature engineering.

By creating an innovative IDS that integrates deep learning and machine learning, this study addresses the inherent security challenges in IoT networks. The main objective is to develop a sophisticated intrusion detection system (IDS) that integrates many ML models and evaluates how well they identify and handle potential security risks in comparison to

the DL model. The two fundamental components of feature engineering—feature selection and feature reduction—make up our approach. We can evaluate the effect of feature engineering on the IDS by measuring the performance of several ML and DL models, both with and without mRMR for feature selection and PCA for dimensionality reduction. Because of this, these feature engineering approaches improve the IDS's performance while drastically cutting down on the amount of time and processing power required to assess the actions of IoT networks. Quickly distinguishing between typical activities and potential security concerns on both a binary (normal/attack) and multi-level (normal/various attack) basis is now achievable with our approach.

1.5 IDS STATE-OF-THE-ART WORKS

Numerous studies have focused on network intrusion detection. For this reason, a more thorough understanding of the field requires the classification and evaluation of different methods. Considering that ML and DL are fields that are always evolving, we examine several models and the methodologies used by them throughout the brief history of these fields.

Deep learning algorithms for cybersecurity were briefly touched upon in [11]. Among the 35 datasets analyzed, the following were categorized: Application traffic on electrical networks, network traffic, Internet traffic, traffic on virtual private networks, application traffic on Android devices, and traffic on Internet of Things devices, and electrical network application traffic. For training and evaluation purposes, many deep learning models are used the Bot-IoT and CSE-CIC-IDS 2018 datasets. We also review the current solutions for intrusion detection based on DL. Neural Networks: Recurrent, Convolutional, Deep Boltzmann, Deep Autoencoder, Restricted Boltzmann, and Deep Neural Networks all seem to exhibit similar kinds of detection results.

Machine learning's practical applications are the focus of this study in several networking domains, including network security, and was published in [12]. Using datasets from the KDD Cup 1999 and the Third International Knowledge Discovery and Data Mining Tools Competition, we analyse intrusion detection methods that are based on anomaly detection, hybrid detection, and abuse detection and NSL-KDD datasets. We take 36 different approaches into account. A lack of real-time implementations, up-to-date datasets, systems tailored to individual requirements, and systems able to detect real-world abnormalities

were also discovered by the researchers. Finally, they present a more complete view of networking and stress the requirement of benchmarking against actual data rather than artificial datasets and predetermined criteria.

One possibility is to use data mining and machine learning to find security flaws researchers are looking into [13]. The approaches that are offered include Decision Trees, Clustering, Artificial Neural Networks, Bayesian Networks, Association Rules, Ensemble Learning, Support Vector Machines, Sequential Pattern Mining, Inductive Learning, Hidden Markov Models, Naïve Bayes, and Evolutionary Computation.

Anomaly detection and misuse-based identification are also considered in these systems. In what follows, we compare and contrast the various methods with respect to their computational complexity and streaming capacities. Among other things, they discuss intrusion detection system (IDS) efficiency, the challenges of contrasting various detection techniques, and more in their final section of the study. Prior to wrapping up, they address IDS performance, the difficulties in comparing various detection methods, model (re)trainability, and provide some suggestions. A review of existing open-source NIDS, publicly accessible datasets and tools, and in a study, the author discusses the security concerns surrounding the Internet of Things (IoT). Additionally, they take into account the outcomes of algorithms that rely on machine learning and have been tested on extensive datasets for network intrusion detection. A crucial component is an IDS dataset containing actual Internet of Things (IoT) intrusions, according to them, and they want to see more emphasis on the semantic links between learning and detection performance.

Among the many cybersecurity issues addressed by the authors of [15] is tackling the issue of network intrusion detection using deep learning. Autoencoders, deep belief networks, convolutional neural networks, recurrent neural networks, and deep neural networks are just a few of the methods they study for network intrusion detection. Recurrent neural networks, autoencoders, and limited Boltzmann machines were the most popular security concerns. They go on to say that the training data's benign-malicious ratio, number of classes, and attack kinds all have a big role in how well an intrusion detection system works. Lastly, they consider the potential outcomes of missed assaults and false alarms, as well as the likelihood that attackers may attempt to bypass security measures. In [16], we see an examination of the ways Deep Learning may be used to intrusion detection via the use of various techniques, such as the identification of spam, phishing, website defacement, and

malware. They preferred research that used generative deep learning techniques for intrusion detection over mixed or discriminative methods. In addition to reviewing the literature, the authors provide a broad framework for using deep learning to cybersecurity. Furthermore, they bring up the possibility of semi-supervised learning in light of the area's accessible unlabeled data. Finally, they stress that deep learning is best reserved for domains with plenty of data, sophisticated nonlinear models, and other characteristics associated with learning.

In, many approaches to anomaly detection are introduced. Their methods for detecting anomalies include rule-based approaches and inappropriate algorithms, such as Support Vector Machines. Furthermore, they evaluate anomaly detection methods based on computing difficulties, output format, and attack priority, and they examine datasets from intrusion detection systems. It seems that this research is only applicable to assaults that occurred during the DARPA/KDD Cup, hence it can't be used with more recent datasets. Research by [18] gives a thorough synopsis of the system. They lay out the basics and then go into detail in a huge table and debate, so their work may be a reference for picking out the right public datasets for the job. Data repositories and traffic generators are among the other data sources that are taken into account. In the end, they make a few conclusions that can be helpful for future research using NIDS datasets. For instance, they say that there will never be a perfect dataset and suggest looking at a number of different ones. Although their study doesn't reveal how it does provide useful data when certain methods or tactics are used.

Research including statistical, knowledge-based, methods based on clustering, ensembles of clusters, fusion, and hybrids of these methods is presented in [19] for the purpose of identifying abnormalities in networks. Along with discussing methods for evaluating detection strategies, they take a look at tools like Wireshark and Nmap that might be used to detect anomalies inside networks. Lastly, along with a list of difficulties, they provide suggestions for detecting anomalies in networks. An article summarises the research on network intrusion detection using Deep Learning and Machine Learning. They go over techniques including k-Nearest Neighbor, SVMs, DBNs, RTNNs, and CNNs, as well as decision trees, RNs, and CNNs. Three points are brought to their attention in this summary: The absence of standard datasets for comparison, inconsistent measures for assessment, and, lastly, the disregard for algorithm efficiency. In addition, they highlight many

tendencies in intrusion detection studies, such as studies on hybrid models, the pros and cons of deep learning, the expanding corpus of literature assessing various methods, and the possibility of future benchmark datasets.

1.6 OBJECTIVES OF THE STUDY

1. Develop a machine learning model to detect anomalous SQL queries.

2. Improve the accuracy of SQL injection attack detection in real-time.

3. Optimize system performance for large-scale SQL databases.

4. Implement feature selection techniques for effective SQL query analysis.

5. Ensure the scalability of the intrusion detection system for varying database loads.

CHAPTER 2

LITERATURE REVIEW

[41] Security across all domains, including databases, networks, and the cloud, has grown in importance in real-time distributed systems since the advent of cyberspace. When faced with evolving threats, present-day intrusion detection systems (IDS) find it difficult to stay involved. The proposed model DR-DBMS (dimensionality reduction in database management systems) combines supervised machine learning methods, dimensionality reduction approaches, and sophisticated rule-based classifiers to enhance the accuracy of intrusion detection for different kinds of assaults. Simulation findings show that the DR-DBMS system effectively used methods for dimensionality reduction and traits selection for determining intrusion assault in 0.07 seconds with a lesser number of characteristics.

[42] presented compressive study on Intrusion detection systems (IDS) are designed to keep an eye out for any breaches in a network's confidentiality, integrity, or availability that may have occurred as a result of malicious or unapproved activity. Machine learning methods for detection and classification, a variety of intrusion detection systems (IDS), and anomaly detection approaches were thoroughly reviewed in this thesis. We provide an architecture for intrusion detection systems (IDS) that combines traditional methods with machine learning techniques. The current detection methods may be enhanced for better attack detection and categorization by including suitable machine learning techniques. Additionally, we have made an effort to evaluate and execute a battery of virtual tests on each machine learning algorithm to compare their performance. For general intrusion detection systems used in industrial control applications, our method offers indications for choosing machine learning techniques.

[43] Emphasizes importances of IDS in Safeguarding sophisticated communication networks requires intrusion detection systems (IDS). Certain patterns, signatures, and rule breaches were the primary targets of these systems' design. Potential new methods to network intrusion detection have emerged Using Deep Learning and Machine Learning methods in the last many years. Techniques like this may tell the difference between regular patterns and those that aren't. Support Vector Machine, J48, Random Forest, and Naïve Bytes were among the machine learning techniques used to assess the Network Intrusion Detection Systems (NIDS) in this article. The methods used binary and multi-class

categorization on the NSL-KDD benchmark data set. Extensive discussion is provided about the outcomes of using such strategies, which exceeded our earlier efforts[43].

[44] Highlighted the emerging challenges faced by Intrusion detection systems (IDS) due to encrypted data, new protocol variety, and an increase in the amount of criminal actions globally. Intrusion detection systems that rely on signatures are now at a stage where inadequate in this case. Several academics have put forth IDSs that use machine learning to detect intrusions previously unseen harmful actions by analysing patterns of activity. Intrusion detection systems that use on machine learning have many advantages than SIDS that rely on signatures are more effective in spotting novel forms of network-based malware. This study evaluated network intrusion detection systems utilising the IDS dataset, which contains the most recent common attacks, and two data resampling techniques in addition to 10 machine learning classifiers. The top three IDS models—XGBoost, KNneighbors, and AdaBoost—perform better than binary-class classification with 99.49%, 99.14%, and 98.75% accuracy, respectively. XGBoost, KNneighbors, and GaussianNB are the most accurate in multi-class classification with 99.30%, 98.88%, and 96.66% accuracy, respectively.

The need of security measures to prevent breaches is growing in direct correlation with the proliferation of internet-connected devices. The purpose of an Intrusion Detection System (IDS) is to identify and block potentially harmful network traffic. The need for anomaly-based intrusion detection systems that use machine learning to identify more recent assaults has arisen because new threats may easily bypass standard signature-based IDS. This thesis will focus on a specific theme: anomaly-based intrusion detection systems that use deep learning. A comparison is made between adversarial machine learning methods and Generative Adversarial Networks (GAN), with conventional deep learning techniques. We use statistical metrics on two separate datasets to assess the strategies. As part of the assessment process, malicious samples are taken into account alongside benign and known attack samples. The last step is to compare the optimal method to other open-source anomaly-based intrusion detection systems. Outperforming all other techniques was a method that used a GAN to generate adversarial samples. Furthermore, when faced with malicious data points, the method matches the performance of current anomaly-based IDS. After reviewing the literature, we have come to the conclusion that intrusion detection

systems based on GANs may be enhanced to better withstand both new and malicious assaults [45].

[46] Emphasizing Network incursions are becoming more common, complex, and dynamic as network sizes are growing at a fast pace. Identifying breaches in a network of this solving the size issue is a significant and challenging task. Recently, there has been a surge in the use of machine learning for network intrusion detection attention lately because of its smart potential. Machine learning-based solutions, particularly those using deep learning, outperform rule-based solutions when it comes to detecting different types of network attacks. Although deep learning has already shown effective in other domains like image recognition and NLP, network intrusion detection is only starting off. Whether it works in practice is an open question, and if so, there are a number of challenges that must be investigated. Two main difficulties in using deep learning techniques to identify network intrusions is the core focus of this thesis. 1) Inadequate training data in the field of network intrusion detection; 2) Excessive human interaction in current machine learning models; 3) A significant number of false positives and poor accuracy in current deep learning solutions. To deal with insufficient data, we suggest a deep learning system that incorporates LightGBM, XGBoost, LSTM, and decision trees; we test it on three datasets to make sure it works. A few popular datasets are the NSL-KDD, UNSW-NB15, and CIC-IDS2017 databases. A model's likelihood of improving the framework's detection is a key factor in its selection. The proposed framework's results show that all four methods routinely beat the best-in-class machine learning-based solutions, proving that the design methods we used for our thesis were effective.

[46] Providing security for such systems has become increasingly important due to the fast rise of information technology in several fields. This advancement in IT has also led to a dramatic rise in the sophistication of cyberattacks. This means that older forms of security software, The NSL-KDD, UNSW-NB15, and ACSS datasets are often used. Because they rely on signatures rather than other data, traditional intrusion detection systems can only respond to known threats by using either preset or user-defined rule sets. When new or altered known assaults don't match the static signature, traditional intrusion detection systems miss them. In conjunction with signature-based intrusion detection systems, using deep learning and other developing behavior-based anomalous activity detection approaches might help reduce false positive and negative rates and make new types of

attacks easier to spot. Deep learning algorithms may learn from patterns and data, making it viable to assess whether traffic is regular or malicious, thereby increasing the detection rate of actual harmful activity. To cut down on false alarms, they may automate the detecting process automatically, eliminating the need for human setting. This thesis aims to investigate how well network flow-based intrusion detection systems that use deep learning methods, especially recurrent neural network designs, perform.

[48] Highlighted the growing importance of protecting sensitive data and networks is quite important. So as to achieve a civilized and informed society, establish a government that is democratic, open, and responsible, and guarantee long-term economic growth, information is playing an essential role. However, corporations are growing more susceptible to cyber-attacks, which are become more complex, dynamic, and damaging, due to their dependency on tech-based systems. The protection of data and computer networks is vital for protecting enterprises from cyber-attacks and minimizing their effect.

There has been promising progress in using machine learning techniques to improve the detection accuracy of intrusion detection systems (IDS). Intrusion detection systems have included several machine learning algorithms. While intrusion detection systems have improved, the majority still have a high false positive and incursion detection rate. The ensemble technique, which combines the predictions of several separate classifiers, was the primary emphasis of this thesis. Different sorts of ensembles and different techniques to integrate the predictions of individual classifiers for an ensemble classifier are considered as we investigate various ensemble methods in the area. The approach is able to use the collective knowledge of its classifiers to improve its performance, even when individual classifiers are poor. This work set out to create and evaluate an ensemble ML model for intrusion detection prediction using the CIC-IDS2017, a standard dataset that the Canadian Institute for Cyber Security has just published. The results of the simulations show that the ensemble model that was suggested achieves an accuracy level of up to 99%, which is far higher than the present IDS systems. Using metrics like recall, F1 score, accuracy, and precision, this algorithm demonstrated encouraging results that bode well for its eventual implementation on actual network infrastructure.

[49] Discussed the adoption of web-based apps has become ubiquitous across all spheres of society, from the private sector to the public sector to the academic world. Most programs are still accessible and operated using web interfaces, even when new technology has

emerged. An architecture for multi-stage log analysis is shown in this study. It integrates machine learning techniques in order to match patterns. In order to detect assaults and help stop them, it utilizes logs produced by the application during assaults. A proof-of-concept prototype is built utilizing pattern matching and Bayes Net for machine learning, and the architecture is presented in depth. According to the results of the experiments, the detection accuracy has been enhanced by combining the two systems into a two-stage system. The multi-stage log analysis technique is very relevant to various intrusion detection applications, and the suggested work is crucial in developing online security. In order to identify different kinds of attacks that impact net banking applications, the suggested system makes use of intrusion detection technologies.

The exponential growth of the Internet and associated communications has caused a meteoric rise in the amount of data sent. Due to their insatiable need for this data, attackers are always devising new methods to steal or manipulate it. The rising prevalence of these attacks poses a serious risk to the security of our systems and is one of the main obstacles faced by intrusion detection systems. By analyzing data sent over a network, intrusion detection systems (IDS) may spot malicious activity. A number of academics have looked at intrusion detection system (IDS) issues and come up with new solutions, but both the detection accuracy and the false alarm rate may be better. Moreover, zero-day assaults are difficult for many IDS to identify. In order machine learning techniques have lately been well-liked by scholars for their ability to detect network intrusions with high accuracy and efficiency. Presented below is a taxonomy of ML techniques in this study together with the idea of IDS. We provide the key criteria for evaluating an IDS and go over several recent examples of IDS that have made use of machine learning, outlining the benefits and drawbacks of each. The next step is to talk about how reliable the findings are from the evaluated work and provide information belonging to the several datasets used for the study. Lastly, we go over some notes, some research obstacles, and some tendencies for the future [150].

Adeyinka Ayodeji Mustapha et.al Examined the increasing vulnerability of e-commerce sites to hackers has grown in tandem with the meteoric rise of online shopping. The databases of these sites typically include sensitive, personal, and even financial information belonging to their millions of users. A very risky sort of attack is SQL injection, which poses a serious threat to backend databases by taking advantage of security holes in online

applications. Although they provide some security, traditional methods of defense such as desktop firewalls, input validation, and parameterized queries are sometimes inadequate when faced with more advanced injection versions or attackers. Machine learning has shown promise as a means to improve cybersecurity in the face of more sophisticated attacks. Presenting a thorough evaluation of the approaches being used to identify SQL injection attacks and how they may affect online storefronts. This study aims to highlight the efficacy of many techniques and real-world uses for bolstering the safety of e-commerce systems by locating and evaluating the relevant literature. To be more precise, two mixed the models (ANN + NB and LR + NB) and five individual methods were evaluated using synthetic and actual datasets. According to the results, Random Forest outperformed other decision tree algorithms because it successfully balanced recall and accuracy. The computational cost of some models and the use of only one dataset were, however, pointed out as drawbacks. This paper describes methods to solve present constraints in future work and gives practitioners ideas on how to pick effective detection models. Increasing the variety of datasets used, improving computing performance, and investigating more complex ensemble techniques and neural network designs are all potential ways to overcome these constraints [51].

Because of attackers take advantage of weaknesses in the database layer, [52] highlighted that SQL injection attacks are a major danger to online applications because they insert harmful SQL code into fields where users enter their data. Unauthorized access, data breaches, and application and database compromise are all possible outcomes of these assaults. Input validation and parameterized queries are two examples of classic approaches against SQL injection, although they aren't without their flaws. These techniques may not be foolproof since they depend on human coding habits. There is an immediate need for sophisticated and automated platforms equipped to identify and forestall SQL injection attacks as attackers are always improving their methods. In this regard, AI demonstrates its value in preventing and detecting SQL injection threats. AI's ability to sift through mountains of data, spot trends, and draw lessons from past assaults makes it a priceless asset in this regard. The use of AI to forecast SQL injection attacks is very beneficial. It is an essential tool because of its anomaly detection, learning from new attack patterns, sophisticated pattern recognition, false positive reduction, real-time protection, and scalability to handle huge applications. Organizations may protect their databases and online apps against SQL injection threats by using AI to improve their defenses.

[53] Emphasized the growing need for innovative solutions to protect their networks and data from ever-changing threat actors, because cyberattacks are becoming more common and more damaging at an exponential pace. It is becoming increasingly difficult to locate a specific security incident or alert due to the exponential growth in the quantity of data produced by security tools and sensors integrated into today's business networks. When it comes to monitoring, preventing, detecting, and responding to cybersecurity incidents and possible assaults on their networks, organizations must depend on innovative tools to aid and supplement human analysts.

Network traffic flow classification as harmless or harmful is the main subject of this thesis. There are two ways in which this work contributes. To begin, NIDSs use supervised learning to identify potential intrusions into a network using a feedforward fully connected Deep Neural Network (DNN). Secondly, when there isn't any labelled hostile data, using unsupervised learning, an autoencoder is used to detect and classify attack traffic. We employ two recently disclosed intrusion detection datasets to build deep neural network models, which address the shortcomings of earlier, more widely used datasets. When trained on these more recent datasets, DNNs demonstrate remarkable effectiveness in supervised learning for detecting and classifying contemporary cyber-attacks that are very accurate, have a low false positive rate, and are very hard to detect. Also, anomaly detection is a good use for autoencoders.

[54] Highlighted that the rapid expansion of online networks has resulted in a steady stream of new threats and assaults. An essential part of any safe network should be the ability to detect intrusions spot suspicious activity or abuse. Recent advances in machine learning have shown beneficial in intrusion detection due to their model-free qualities, which allow them to recognize network patterns and determine whether they are harmful or not. A large number of machine learning models were used in intrusion detection systems that are based on anomalies. The research community is showing a marked increase in the need for trustworthy and realistic assault datasets. This study carries out an in-depth examination of the most current dataset, CICIDS2017. The investigation uncovered several issues and shortcomings within the dataset. To address these issues and provide an optimal CICIDS2017 dataset, a few methods are put up. During the study, a total of 36 features were recovered, which is more than the 23 features taken from the dataset derived from

literature. The collection of 36 characteristics outperformed the others while analysing testing model losses, accuracy, and F1-score metrics.

Cybersecurity is an essential field of study because of the pervasiveness and importance of networks in today's society explained by [55]. An important part of cyber defense is surveillance technology that monitors the safety of all the network's software and hardware. Problems in detecting new assaults, lowering the false alarm rate, and improving detection accuracy persist in current IDSs, despite decades of progress. In an effort to solve these problems, several researchers have focused on developing intrusion detection systems (IDSs) that use machine learning approaches. Automatically and with high accuracy, machine learning algorithms can differentiate between normal and unusual data. The high generalizability of machine learning algorithms further increases their potential to identify previously unseen threats. An area of intense study, one branch of machine learning is known as deep learning known for its exceptional performance. For the purpose of categorizing and summarizing IDS literature this research used deep learning and machine learning suggests a taxonomy that uses data objects as the primary dimension. Cyber security academics might benefit from this categorization system, in our opinion. The idea and classification of IDSs are first defined in the survey. The following section presents the popular IDSs use machine learning methods for detecting intrusions, metrics, and benchmark datasets. Afterwards, we use the suggested taxonomic system and the representative literature as a starting point to describe how to use deep learning and machine learning to address critical issues with intrusion detection systems. Lastly, by looking at recent representative research, we may talk about the difficulties and potential future advances.

Through the analysis of suspicious behaviors and the monitoring of actual network traffic, A lot of research has focused on NIDS, or network intrusion detection systems. But NIDS can't always catch all kinds of assaults; Advanced Persistent Threats (APT) are only one example. Furthermore, NIDS is unable to see full traffic data because of authorization issues or encrypted communication. In order to circumvent these restrictions, the logs, files, and directories of a host are scanned by a host-based intrusion detection system (HIDS) for signs of advanced persistent threats (APTs), which often inject infected nodes with harmful files. To enhance intrusion detection performance, this research proposes a system that blends NIDS and HIDS. It flows via a BERT model to deal with host data in the same way

as machine learning models deal with data about network flows that is based on Natural Language Processing (NLP). This model converts the host data from between a numerical representation and a verbal one. In order for sophisticated Machine Learning (ML) models to analyze features based on network flows and hosts, in order to transform the two-dimensional host-based attributes provided by BERT into one-dimensional vectors, the feature flattening method is used. A two-stage collaborative classifier is used to enhance HIDS's effectiveness. When it comes to detecting network intrusions, this classifier uses multi-class and binary machine learning techniques. A binary classifier was used to identify innocuous samples, simplifying the original problem. Then, a multi-class supervised learner was used is used to categorize the attack data and determine the kinds of attacks. Therefore, compared to the baseline classifier, XGBoost, the two-stage collaborative model performs better overall. We demonstrate that the suggested approach generalizes on two popular datasets, namely CICIDS 2018 and NDSec-1. As an example of traditional ML, we test how well XGBoost performs. Using the CICIDS 2018 dataset, there is an 8.1% improvement in attack detection performance, as measured by the macro average F1 score, and by 3.7% under the NDSec-1 dataset when host and network information are included. At the same time, when contrasted with conventional ML models, collaboration classifier with two stages achieves significant gains in detection performance across the board, particularly for DoS-LOIC-UDP (a 30.7% improvement) and DoS-Slow HTTP Test (an 84.3% improvement) [56].

In light of the ever-increasing data quantities, the importance of Big Data analysis and information security systems has lately changed. Intrusion detection systems (IDS) are set up to is to keep an eye on data and identify any attempts at system or network penetration [30]. The massive volume, variety, and velocity of data generated by the network have made the data analysis process of detecting attacks using conventional methods very difficult. For a more precise and time-saving data processing procedure, IDS employs Big Data methodologies to handle Big Data. An intrusion detection model called Spark-Chi-SVM was presented in this article. Our intrusion detection model was built on top of the Apache Spark Big Data platform with features selected using Chis Selector. At its core, the model is a support vector machine (SVM) classifier. In order to train and evaluate the model, KDD99 was used. Experiment results were compared between the Chi-SVM and Chi-Logistic Regression classifiers. Research results shown that the Spark-Chi-SVM model is efficient for Big Data, has good performance, and minimizes training time.

[58] Explained that nowadays, data is flowing at a rate of gigabytes per second, making intrusion detection a laborious and slow procedure. As a result, intrusion detection systems need a better and more effective detection method to identify malicious actions. Constraints on performance, sustainability, security, privacy, energy use, fault tolerance, scalability, and adaptability are some of the issues that arise when dealing with the availability, complexity, size, and complexity of big data. These issues also affect the design and management of IDSs and IPSs, which stand for intrusion protection systems. To ensure company computer networks are secure, accessible, and reliable, IDSs and IPSs use distinct methodologies in their respective detection and prevention of intrusions. Intrusion detection methods for distributed systems in large data are the subject of this paper's in-depth analysis, which also details methods for preventing intrusions and detecting them using machine learning and big data analytics.

The transition from restricted local networks to remotely accessible networks is happening quickly in industrial control systems (ICS)highlighted by [59]. since of this change, robust an intrusion detection system and cybersecurity anomaly detector are necessary for these systems. However, since ICSs are complex and diverse, developers are currently working on creating systems that are both well-defined and dependable. Potentially applicable to all ICS, used machine learning methods for intrusion and anomaly detection at the network level might provide comprehensive security. In order to categorize this paper explores two ML applications using the assault label from the UNSW-NB15 dataset. A standard dataset, the UNSW-NB15 was built using and includes labels for common behaviors and attack vectors generic network communications. Because it is mathematically simple, K-Nearest Neighbors (KNN) was used to build a baseline. Because of its reusability and potential application in production settings, A Multi-Layer Perceptron (MLP), a feed-forward artificial neural network, was used for the purpose of comparison after the baseline was established. Although both KNN and MLPs still need more refinement and improvement before being used on a real-world production scale, the experimental findings demonstrate that they are viable ways for recognizing fraudulent network traffic.

[102] In order to make computer networks more secure, researchers have extensively examined and used intrusion detection systems. Recently, there has been a lot of buzz about how machine learning and intrusion detection systems might work together to improve cybersecurity due to the rising assault volume and the tremendous advancements in

machine learning. Training a machine learning model often requires access to massive amounts of data samples. For the reason that the machine learning model's training and detection Input data that is too big may cause performance issues. To eliminate superfluous or unneeded characteristics from a dataset, feature selection is an essential approach. This research used a feature selection strategy to determine which attributes are most useful for a network intrusion detection system machine learning model by combining attack-specific characteristics with sophisticated feature selection methods. For the purpose of creating the ideal feature subset, our benchmark dataset of choice was the most current one, CSE-CIC-IDS2018., which has extensive attack variety and characteristics. Machine learning models utilizing a decision tree classifier were used to generate the experimental results, which were then evaluated according to f1 score, recall, accuracy, and precision.

[61] Highlighted For today's technologically advanced society, cyber defense is of paramount importance. The increasing sophistication of many assaults calls for more robust vulnerability management, analysis, and identification procedures. Systems act as a primary safeguard by should include intrusion detection and prevention systems. The article gives a summary of many factors that should be taken into account while developing machine learning-based intrusion detection systems. The classification scheme for IDSs, the types of attacks they detect, and the organizational framework for responding to security events are all covered in this examination. The report goes on to outline the databases utilized in various investigations that employed Machine Learning to identify breaches. In addition, the most often used measures for evaluating algorithm performance are detailed. Lastly, we will go over the obstacles that will drive more study.

Highlighted the concern related privacy issues involving sensitive information, such as medical records and bank account details, have arisen in response to the deluge of data generated by the proliferation of smart gadgets [62]. This information is always moving over the network and can meet other users' requests for it. This data transfer may be completely legitimate or it may be the result of an intrusion designed to introduce malicious data into the network. Attack signature patterns are what firewalls and older forms of intrusion detection systems use to identify intrusions. But it won't be enough to catch sophisticated or undiscovered assaults. The employment of intelligent approaches is crucial for the detection of many unknown attack kinds. Several machine learning methods that have been suggested in the last several years are examined in this article. In order to identify

suspicious activity in network data, this research used many categories. The models were developed and assessed using the 2017 dataset from Intrusion Detection System research at the Canadian Institute for Cybersecurity (CIC-IDS-2017). This compilation of data encompasses both recent and past intrusions. The following methods were used in the experiments: decision tree, logistic regression, adaptive boosting, Ensemble methods, as well as gaussian naïve bayes. To evaluate the models, a battery of tests was administered. The recall, accuracy, precision, F1-score, false positive rate, receiver operating characteristic curve, and calibration curve were all metrics that were considered.

Most of the Cyber Physical System (CPS) uses User behavior due to infrequent breakdowns or cyber assaults may significantly impact the expansion of security, which is of the utmost importance [63]. Protecting the CPS is of the utmost importance, as mentioned in the rules. Security software such as firewalls, intrusion detection systems (IDS), and antivirus programs integrated security solutions that are now commonplace in many networks. Firewalls filter incoming network traffic according to its physical address or location. Firewall policies alter data transmissions across networks. Only the number of states and the hosts that are receiving material are known to the firewall. One kind of intrusion detection system is the security appliance, which keeps an eye on all of the connections to the network, checks for any suspicious behavior, and then notifies the system administrator or system user. Here we take a look at SQL injection as it pertains to the HTTP/HTTPS protocol and how it may be used to get past web application firewalls (WAFs) and steal sensitive data. One kind of injection attack is SQL injection, which allows hackers to access a system and execute harmful applications. There is a chance that data will be erased or stolen as most people who are affected aren't prepared to use these tactics. If this happens, the hacker has the power to access and alter the data, which might compromise its availability, integrity, and confidentiality. The main aim of this article is to look at how machine learning and deep learning may improve the detection of SQL injection prevention. We hope that by providing this thorough overview, we can keep researchers informed and help them better grasp how intelligence and SQL injection protection connect.

Web apps are essential in today's world. Every day, more and more people are using online applications. Databases are used by the majority of companies to store user data or information about the services they provide. When interacting with databases, Structured Query Language is often used. The hacker inserts harmful SQL statements into through the

database in an attempt to inject SQL queries. There is a significant security risk with SQL injection since it may change database values, delete the whole database, or steal database material. By utilizing SQL injection, an attacker may get unauthorized access to the whole database. When a web application doesn't have proper validation or filtering in place, SQL Injection attacks might happen. Machine learning methods including Convolution Neural Networks, Decision Trees, Support Vector Machines, Gradient Boosting, and Naive Bayes for the detection and analysis of SQL injection threats [64].

The allure of e-commerce sites to hackers has grown in tandem with the meteoric rise of online shopping. The databases of these sites typically include sensitive, personal, and even financial information belonging to their millions of users. A very risky sort of attack is SQL injection, which poses a serious threat to backend databases by taking advantage of security holes in online applications. Although they provide some security, traditional methods of defense such as desktop firewalls, input validation, and parameterized queries are sometimes inadequate when faced with more advanced injection versions or attackers. Machine learning has shown promise as a means to improve cybersecurity in the face of more sophisticated attacks. An exhaustive evaluation of the existing techniques for detecting SQL injection is the goal of this study attacks and how they may affect online storefronts. This study aims to highlight the efficacy of various algorithms, and how they may be used to make e-commerce systems safer by locating and evaluating the relevant literature [65].

[66] Explains the Network security requires good intrusion detection due to the increasing interconnectedness of computers, sensors, and servers. This study delves into e-commerce platforms' host-based intrusion detection systems that use machine learning techniques. Using publicly accessible datasets such as CSIC 2010, we provide a machine learning method for identifying fraudulent or benign HTTP requests. A variety of statistical methods, including logistic regression, k-means clustering, probability mixtures, and random forests, MLP classifier, KNN, decision tree, AdaBoost, XGBoost, and LightGBM are among the techniques that we analyze, with an emphasis on feature selection. To get the best results, we use visualization tools for understanding the models' parameters and hyperparameter tweaking for adjusting the class imbalance. This study demonstrates how supervised and unsupervised Machine learning has the potential to enhance intrusion detection systems and network security.

Since its inception for scholarly and military research, the internet has grown to accommodate the needs of its ever-increasing user base, many of whom have interests outside of these original domains. Upholding the network's security became an increasingly pressing concern as its breadth grew[67]. Due to its diverse user base and network architecture, the internet must be kept as secure as possible to ensure the transfer of sensitive information is completely safe. Several anomalies have the potential to breach private networks. Recent years have seen a plethora of literature on the topic of network security; this study aims to contribute to that literature by delving further into these attacks, suggesting effective ways to identify network intrusions, and presenting an ensemble classifier, which combines three distinct machine learning algorithms. In order to identify remote-to-local (R2L) assaults, an ensemble classifier is used. When testing the network dataset with individual machine learning models, the accuracy level is lowest, but with the ensemble classifier, the total efficiency is 99.8 percent.

A database has been an essential means of storing information for several public and commercial sectors, including enterprises, banks, and educational institutions. The widespread usage of this cloud-based database has prompted several scholars to concentrate on privacy and integrity as key security concerns. Researchers have suggested security procedures to identify suspicious database user activity using an Intrusion Detection System (IDS). But new research didn't account for transactional SQL queries that include nested queries depending on roles. The Multiplet data structure is suggested in this study as a means of discovering abnormalities in layered transactional SQL queries. The given data structure takes these queries and uses them to build the user's profile, which allows it to identify intrusions. One such approach may be to compare the transaction log with the SQL statements in an effort to improve the detection accuracy supplied by the user using this Multiplet. In the event that the algorithm does not find a correlation between the normal user inquiries and the suspiciously sent questions, it will notify the user of the possible invasion. According to the findings of the Naive Bayes Classifier (NBC), an intrusion was successfully identified with a 97% accuracy rate. The paper's practical significance is further shown by the detection of a sophisticated transactional database that often departs from the typical user SQL queries [68].

The discipline of cybersecurity has grown in importance inside the digital realm in the last few years. The management and organization of network architecture has been radically

altered by the proliferation of networks. But the danger of cyberattacks has grown nowadays, as a result of the ever-increasing complexity of modern networks and the explosion in the number of linked devices. Opportunities and problems are both brought about by the continuous advancement of technology. Keeping networks and sensitive data safe from hackers and other bad actors is a huge obstacle. A more adaptable and programmable architecture, Software Defined Network (SDN) has grown into a standard component of modern networks. Because SDN networks are growing in popularity and appealing to clients as targets owing to their programmability and dynamic nature, researchers should concentrate on identifying attacks in SDN. Without adequate security measures, the centralized controller, which is the foundation of a software-defined network (SDN), becomes a weak spot and a desirable target for cybercriminals. To lessen these dangers, researchers should focus on threat detection in SDN. Researchers can create effective detection systems and countermeasures by learning about the possible vulnerabilities and attack tactics that are unique to SDN. The network is better protected from possible threats and the effects of successful assaults are lessened by using this technology. Therefore, to fully understand SDN security problems, this survey article reviews Intrusion Detection Systems (IDSs) in SDN [69].

[70] explains different threats to the security of today's wireless networks are growing in number and sophistication at a dizzying rate. Network security is a growing concern due to the fact that computer networks and applications are constructed on many platforms. Operating programs, whether simple or complicated, may include security flaws. In this context, "intrusion" is an effort to compromise availability, completeness, or security. Using an intrusion detection system, network security weaknesses and anomalies may be detected. Despite its often-cited shortcomings as an all-encompassing strategy for thwarting invasions, the area of intrusion detection technology development has been rapidly expanding. It is also a top responsibility for network administrators and security professionals. More secure systems will not be able to fully replace it because of this. Based on past intrusion detections, intrusion detection systems (IDS) may use data mining to foretell when more incursions will occur. This study presents a comprehensive literature assessment on IDS using data mining methodologies. Using both real-time and benchmark datasets, we will first examine methods for detecting intrusions via data mining. Throughout this study, we evaluate and contrast several strategies for detecting network

intrusions, along with their advantages and disadvantages. We provide methods to enhance network intrusion detection in this article.

There is a great deal of private and sensitive data flowing due to the use of technology by various devices. Computer security solutions, such for this reason, IDSs, or intrusion detection systems, are crucial. Our intrusion detection system (IDS) here can analyze data using the NetFlow protocol. During the first phase of our research, three distinct kinds of assaults were correctly identified using an examination of previously gathered flows. The second stage included preparing the flows for use in ML algorithms. Subjects—NetFlow, machine learning, intrusion detection system[71].

When thinking about how to keep sensitive data safe from prying eyes, security should be your first priority. In addition, cybercriminals are constantly releasing new types of assaults that make it impossible for users to control their computer systems or networks. Therefore, it is very important to see more cybersecurity research into topics like intrusion detection and prevention systems. One method that has proven successful in the fight against malicious assaults is the intrusion detection system (IDS). Seven different ML algorithms have been tested in this study utilizing the CICIDS2017 intrusion detection dataset. The next step was to evaluate the chosen algorithms using a number of performance indicators. K-Nearest Neighbors (KNN) classifier outperformed competing machine learning classifiers in recall, accuracy, precision, and F1-score, according to the experimental results. To be fair, KNN was the only machine learning classifier that took too long to train its models[72].

[73] stated that the primary function of intrusion detection systems (IDS) to monitor how well a network is doing and to look into any indications of concerning behavior that could be happening on the network. Recently, raising concerns about potential network activity by intrusion detection systems. This is because machine learning has shown to be able to learn and adapt, while also enabling a rapid reaction. An approach to intrusion this study presents a method for detection and classification using machine learning. The model begins by obtaining and properly formatting the data set. Then, it uses feature selection to identify which features are worth considering. Then, the KNIME platform, which is where the classifiers are applied, processed the cleaned-up data set. We used three separate classifiers to improve performance and get a good comparison. The KNIME Analytics Platform has been used to run and evaluate the predicted classifiers on the (CICIDS2017)

datasets. The paper compares and contrasts the three classifiers—SVM, RProp, and Decision Tree—and their respective applications. In addition, different findings from related investigations were compared to the ones that were gathered. The trial findings demonstrated an average accuracy rate of 90.59% and a maximum of 98.6%, which was satisfactory when compared to other methods. This study's statistical findings encourage future studies in the area to use machine learning for data analysis and cyber security purposes, ultimately leading to more precise intrusion detection systems.

[74] Discussed the rapid evolution of the Internet is evolving throughout time, computer networks, and the programs that make use of them continue to expand at an exponential rate. Due to the high potential for assault and the severity of the consequences, several Monitoring is done via the use of Intrusion Prevention Systems (IPS) and Intrusion Detection Systems (IDS) networks for intrusions and protect them from harm. These intrusion detection technologies aren't offering foolproof solutions or accurate deployments since there aren't enough datasets. This study presents a thorough review of the network properties and outlines the problem that pertains to CICIDS Datasets. A sophisticated Security System Monitoring This research also presents the system architecture. It can detect harmful or benign system activity and identify all network traffic. The use of deep learning software, the suggested system can assess various real-world models, leading to more accurate results.

These days, rapid growth of doing business and communicating with others is easier and more commonplace than ever before thanks to the internet emphasized by [75]. Users upload millions of data points over different online channels every day, and some of that data can be harmful. Hence, web application security is crucial. They are prone to several vulnerabilities that, if ignored, might cause damage due to their easy accessibility. In order to do numerous illicit acts, the intruders use these weaknesses to gain illicit entry. With all the many ways it might manifest, SQL Injection is one kind of attack that is easy to launch but difficult to detect. Loss of property, exposure of sensitive information, or theft could occur as a consequence of this. This research has examined the current problem-solving tools, including AMNESIA and SQL Rand, and has assessed their strengths and weaknesses. To identify SQL Injection attacks, we developed a classifier. To identify threats, the suggested classifier uses a mix of an approach for Role Based Access Control is included into the Naïve Bayes algorithm. Each of the three SQL injection attacks

(comments, union, and tautology) is utilized as a test case to evaluate the proposed paradigm using.

Many new difficulties and dangers to information security, including assaults, digital infiltration, data theft, etc., have emerged due to the rapid growth of modern computer networks. With the advancement of AI, intrusion detection systems (IDS) have become more crucial for ensuring the security of data. Among the latest crop of machine learning-based intrusion detection systems are support vector machines (SVMs), recurrent neural networks (RNNs), long short-term memories (LSTMs), convolutional neural networks (CNNs), deep neural networks (DNNs), Boltzmann machines (BMs), decision trees (DTs), stack autoencoders (SAEs), and random forests (RFs) with deep neural networks. Also included is an overview of the network-based datasets and our thoughts on them. The evaluation also highlights the benefits of each dataset[40].

[39] Highlighted that one of the most significant research applications challenges the realm of information security known as intrusion detection systems (IDS). It is becoming more difficult to enhance conventional IDS methods due to the rising frequency of sophisticated network assaults. Data analysis, protection, and identification must be done efficiently. This study proposes a thorough overview of how IDS may be enhanced using Applying ML and DL methods to determine lower mistake rates and improve detection accuracy. An analysis and summary of the most current research articles published from 2018 to 2021 in the area of IDS applications of ML and DL are given. The following are the four primary areas of analysis: First, ideas and taxonomy of IDS. (2) ML and DL approaches' strengths and shortcomings. thirdly, datasets used as benchmarks for intrusion detection systems. (4) A comprehensive review of recent articles that have improved IDS using ML and DL, highlighting the strengths and weaknesses of each research. An approach to integrating ML and DL into the IDS is suggested after reviewing the relevant literature.

[150] Highlighted that the Internet and associated communications have grown at an exponential rate, the amount of data sent has also increased dramatically. Because attackers need this data so much, they are always devising new methods to steal it or change it. Threatening the security of our systems, the rising frequency of these attacks is one of intrusion detection's biggest challenges. By analyzing data sent over a network, intrusion detection systems (IDS) may spot malicious activity. There is still potential for advancement in detection accuracy and false alarm rate, even though several researchers

have explored and created innovative intrusion detection system (IDS) systems. Moreover, zero-day assaults are difficult for many IDS to identify. In order machine learning algorithms have lately been popular among academics for their ability to swiftly and effectively detect network infiltration. Presented here is a taxonomy of ML techniques this study together with the idea of IDS. We provide the key criteria for evaluating an IDS and go over several recent examples of IDS that have made use of machine learning, outlining the benefits and drawbacks of each. The next step is to talk about how reliable the findings are from the evaluated work and provide information belonging to the several datasets used for the study. Finally, the results, challenges, and future directions of the area are discussed.

[38] Discussed increasing the prevalence of hacking and other forms of illegal activity has skyrocketed with the expansion of people's reliance on information technology. Security flaws in software-defined network controllers make sensitive company information and other resources susceptible to hacking. When it comes to keeping computer networks secure, an intrusion detection system that utilizes models for machine learning is of paramount importance. There are a number of network systems that detect intrusions using deep learning and machine learning. The literature study revealed that several publications pertaining to intrusion detection have been written with a focus on IDSs that use machine learning. Modern intrusion detection systems do have several limitations, however, including a slow processing rate, an inadequate detection rate, a lengthy training period, and a relatively large FAR. The majority of SDN intrusion detection systems rely on signatures, which can only identify previously seen assaults and need constant database updates to identify newly introduced threats to SDN network controllers. Our solution to these problems is an intrusion detection system that uses machine learning to analyse data flows. This suggested research set out to improve intrusion detection model performance by reducing training time, increasing detection rates, and enhancing overall performance. To do this, the researchers planned to employ parallel training and choose the most effective parameters for the models. Incorporating the NSL-high dimensional KDD's abstract feature representation into the model was achieved via the use of several hidden layers. Comparing CNN and its variant architectures against many traditional ML classifiers, they have done quite well. Experiment results showed that the proposed deep learning-based anomaly intrusion detection system for software defined networks had better detection accuracy (99.61%), ROC (99%), training time, model loss (low), reconstruction error (low), false-positive rate (low), and overfitting (low) than competing models. How well a software-

defined network's anomalous intrusion detection system based on flow-based learning and deep learning works has been validated via extensive experimental testing.

[51] Examined the increasing vulnerability of e-commerce sites to hackers has grown in tandem with the meteoric rise of online shopping. The databases of these sites typically include sensitive, personal, and even financial information belonging to their millions of users. A very risky sort of attack is SQL injection, which poses a serious threat to backend databases by taking advantage of security holes in online applications. Although they provide some security, traditional methods of defense such as desktop firewalls, input validation, and parameterized queries are sometimes inadequate when faced with more advanced injection versions or attackers. Machine learning has shown promise as a means to improve cybersecurity in the face of more sophisticated attacks. An exhaustive evaluation of the existing techniques for detecting SQL injection is the goal of this study attacks and how they may affect online storefronts. This study aims to highlight the efficacy of various algorithms, and how they may be used to make e-commerce systems safer by locating and evaluating the relevant literature. To be more precise, two mixed the models (ANN + NB and LR + NB) and five individual methods were evaluated using synthetic and actual datasets. According to the results, Random Forest outperformed other decision tree algorithms because it successfully balanced recall and accuracy. The computational cost of some models and the use of only one dataset were, however, pointed out as drawbacks. This paper describes methods to solve present constraints in future work and gives practitioners ideas on how to pick effective detection models. Increasing the variety of datasets used, improving the speed of computing, and investigating more sophisticated ensemble techniques and neural network designs are all potential ways to overcome these constraints[51].

[36] emphasized the critical role of Information management systems that store data, the most precious asset of any business. When it comes to attacks on computer systems or networks, traditional intrusion detection systems are more suited and are unable to identify attacks that target databases specifically. Consequently, intrusion detection systems must be included inside the repository. Attacks may be detected by our proposed intrusion detection system both at the database level and at the user task level (between transactions) transaction level. To achieve this goal, we provide a transaction-level detection approach that relies on specifying the anticipated database application transactions. Next, we suggest

a data-mining-based anomaly detection approach at the inter-transaction level. This approach makes use of rules and patterns found over time. This system can detect fraudulent activity on both the transaction and inter-transaction levels using a hybrid technique that combines specification-based detection with anomaly detection. This is an advantage over earlier database intrusion detection systems. We conducted various tests to test how well the suggested system worked. The proposed method is both accurate and effective, according to the experimental evaluation results.

[31] It is growing more difficult to identify intrusions precisely as cyber-attacks advance in sophistication. If the attacks are not prevented, data accessibility, integrity, and reliability as well as the trustworthiness of security services confidentiality, might be compromised. Multiple intrusion detection systems have been proposed in the literature to deal with computer security issues; two examples are Signature-based Intrusion Detection Systems (SIDS) and Anomaly-based Intrusion Detection Systems (AIDS). In addition to a dataset survey and a summary of key recent articles, this survey research also offers a taxonomy of contemporary IDS often utilized for assessment. In order to make computer systems more safe, the text also delves into the difficulties that lie ahead in terms of research into combating attackers' evasive methods and outlines these solutions.

[34] Discussed the increasing prevalence of attacks on computer networks, also known as "cyber-attacks," are becoming the norm and impact almost all devices linked to the Internet every single day. Because these threats may be effectively countered with the use of deep learning and machine learning without requiring human interaction, organizations are increasingly turning to these technologies. The greatest benefit of machine learning is its inherent capacity to identify, mitigate, prevent, recover, and handle untrained attack types autonomously, without the need for explicit programming. In addition to explaining the many forms of cyberattacks, this study will demonstrate the wide variety of techniques used to counter them. Data on the categorization algorithms, how they work, how accurate they are, and how long it takes to test them are all provided. Various algorithms were used in this experiment, including the following: Algorithms such as the Logistic Regression, Gaussian Naïve-Bayes, Support Vector Machine (SVM), Stochastic Gradient Descent, Decision Tree, Random Forest, Gradient Boosting, K-Nearest Neighbor, Artificial Neural Network (ANN) (which also included the Multilevel Perceptron algorithm), Convolutional Neural Network (CNN), and Recurrent Neural Network (RNN) algorithms exist. Research

indicated that decision tree classifiers were among the several machine learning methods that were investigated and logistic regression required the least amount of time to develop and achieved an accuracy of more than 90% when it came to detecting malware in different test datasets. Despite being easy to build, the accuracy that the Gaussian Naïve-Bayes classifier managed to get was just 51-88%. The development of Gradient Boosting, Multilevel Perceptron, and non-linear SVM algorithms took an extremely lengthy time. When it came to accuracy, the Random Forest Classification algorithm was the clear winner.

[33] Highlighted that the intrusion detection system is one tool used to prevent harmful attacks. In addition, the methods and tools used by attackers are always evolving. On the other hand, it is not an easy operation to install a well-recognized IDS system. Using the KDD intrusion dataset, this study conducts and analyses a battery of tests to rank different machine learning classifiers. In order to assess the chosen classifiers, it was able to calculate many performance measures. Performance metrics for false positives and false negatives were prioritized in an effort to enhance the intrusion detection system's detection rate. Research showed that random forest classifiers had the highest average accuracy rates and decision table classifiers the lowest false negative rates.

[32] Emphasized that the database contains the majority of an organization's important information resources; safeguarding this information from unauthorized access is a weighty issue. Traditional security measures, however, aren't built to identify unusual behavior from database staff. The best way to protect databases from hackers is using an IDS, which provides an extra layer of security that isn't achievable with integrated security solutions. This work presents a technique for RBAC database anomaly detection using a hexplet data format, which condenses raw transactional SQL queries. By removing the user's personal details, this structure may simulate typical database access behavior and detect imposters. With the help of the data found in the transaction log, this hexplet keeps the correlation between SQL statements inside the same transaction intact. Our goal is to increase detection accuracy, particularly for suspicious activity found inside the business. By using the most fundamental method of supervised learning for profile creation and transaction legitimacy evaluation, the model employs naive Bayes classifier (NBC). The results of the experiments show that when the suggested model in relation to the detection rate.

[82] Discussed that smarter everything is the result of increased demand across several industries driven by technological advancements. Although it is susceptible to assaults and threats, several industrial or manufacturing businesses, healthcare sectors, smart retail, agriculture, and more have benefited from the Internet of Things (IoT), which connects everything via networked nodes. Consequently, making safeguarding the IoT necessitates privacy and security measures. In this article, we'll look at the ways in which IoT and IDS security might be enhanced with the use of machine learning and deep learning capabilities and its use in IoT networks. Along with the problems and difficulties that make up the IoT, this piece goes on to discuss the various Intrusion Detection datasets.

[30] Discussed In light of the ever-increasing data quantities, the importance of Big Data analysis and information security systems has lately changed. The purpose of an intrusion detection system (IDS) is to keep an eye on data and identify any attempts at system or network penetration. The massive volume, variety, and velocity of data generated by the network have made the data analysis process of detecting attacks using conventional methods very difficult. For a more precise and time-saving data processing procedure, IDS employs Big Data methodologies to handle Big Data. An intrusion detection model called Spark-Chi-SVM was presented in this article. We used Apache Spark, a big data platform, to build an intrusion detection model, and Chis Selector was used to choose features. A classifier called a support vector machine (SVM) was used to create the model. We used KDD99 for both the training and testing of the model. In this experiment, we tested two classifiers: Chi-SVM and Chi-Logistic Regression. We found that the Spark-Chi-SVM approach works well with Big Data in our experiments, has a short training period, and produces good performance.

[29] Discussed the rapid advancement of technologies such as AI, cloud computing, enormous data generated by the web, and its highly regarded useful uses including distributed data in both symmetric and asymmetrical ways brought about several beneficial changes in our lives. But it's also a reality of life in the age of catastrophic, ever-increasing cyberattacks. Therefore, it is imperative that researchers develop and deploy intrusion detection systems (IDSs) that use state-of-the-art machine learning skills in order to lessen the impact of these regrettable cyber dangers. However, reliable and efficient IDSs are difficult to achieve owing to issues like excessive model building and testing time, high false alarm rates (FAR), inefficient use of labelled training datasets, and low accuracy

caused by numerous irrelevant and redundant features. Individual machine learning classifiers also fail to detect all novel attack types. Consequently, this study put up an encouraging using an ensemble classifier as part of a hybrid feature selection (HFS) approach. This method effectively chooses pertinent features and delivers consistent attack categorization. In the first step, we combine CfsSubsetEval, a rule-based engine, and genetic search for efficient selection of highly correlated feature subsets. This increases the generalizability of learning algorithms—attributes of symmetry learning—and drastically simplifies the model. Another new ensemble classifier we provide, KODE, classifies the asymmetry probability distributions of benign and harmful events in a consistent manner. This classifier takes an average of probabilities and votes on them. Expectation-Maximization, DBSCAN, K-means, and One-Class SVM are all part of it. In conjunction with many datasets and metrics, such as UNSW-NB15, NSL-KDD, and CIC-IDS2017, and 10-fold cross-validation, HFS-KODE produces outstanding results. With a detection rate of 99.75% for CIC-IDS2017, 96.64% Outperforming state-of-the-art feature selection, specific current intrusion detection systems (IDSs) approaches, and individual classification algorithms, it achieved a remarkable 99.93% accuracy for UNSW-NB15 and 11, 8, and 13 relevant features, respectively, in the aforementioned datasets. Three separate measures of performance accuracy were 99.99%, 99.73%, and 99.997%. Last but not least, HFS-KODE stands out when considering the much reduced FAR and time, as well as the removal of the need for labelled datasets outperforms several existing methods.

[28] Highlighted that protecting computer systems from diverse different types of cyberattacks has been a major focus for many groups, including academics, government agencies, businesses, and individuals using wired and wireless networks. But in the last ten years, the data that every device—and servers in particular—handles have grown at an exponential rate, that the safety of these gadgets must be given top priority. This essay focusses on the challenges of applying machine learning for intrusion detection. In this article, we will examine some of the challenges associated with developing and using machine learning for intrusion detection. With the use of a comparison of different ML algorithms on diverse datasets and the summarization of their performance using multiple performance measures, we also aim to determine the optimal technical solution for changing use patterns. Intrusion detection in dynamic settings involving several network technologies is the focus of this article, which also discusses current research issues and future directions in the field.

[27] Cybersecurity analysts always seek greater precision notwithstanding the wealth of literature on intrusion detection systems, and worldwide threat assessments to protect their cyber infrastructure in both academia and business. This enormous data set is challenging to track and analyze utilizing conventional methods, and big data represents the major obstacle for intrusion detection systems. Applying Big Data with a minimal learning curve and a high precision is now possible thanks to deep learning, a new method that has just evolved. In order to identify various assaults in their early stages, using big data and deep learning techniques, we provide an intrusion detection system (IDS) solution that runs on the cloud. We create the proposed system on Microsoft Azure Cloud and demonstrate its efficacy using a convolutional neural network (CNN-IDS) in combination with the distributed computing environment Apache Spark and the Keras Deep Learning Library. Processing power and storage capacity are both provided by Azure Cloud. Using the CSE-CIC-IDS2018 dataset, we assess the model's performance across two classification domains: binary and multiclass. The combination of the Apache Spark engine with deep learning techniques allowed our system to function very well.

[26] Highlighted that with the rapid expansion of the internet such a fast pace, intrusion detection emerged as a key area of study for computer security researchers. Data mining methods are used by several intrusion detection systems (IDS) to categorize data pertaining to network traffic and identify various security breaches. We highlight some of the datasets and methodologies used for network anomaly detection in this study. We take a look at the most recent studies that have attempted to build network intrusion detection systems and evaluate various machine learning approaches. Additionally, we provide a synopsis of several deep learning methods and how they may be applied for intrusion detection systems. Giving a researcher an idea of where things stand in this area of study is the main goal of this survey.

[25] Discussed services provided by the cloud have become more important to people's daily lives. All sorts of people and organizations, from sole proprietors to huge corporations, rely on cloud services for a lot of their daily operations. It has made it possible for customers to access businesses' services whenever and whenever they want, at the lowest possible price, all over the Internet. Cloud networks are susceptible to several forms of assaults, notwithstanding their benefits. The hazards connected with cloud services, however, have grown in tandem with their rapid use. The backbone of any cloud

architecture is its networks, thus keeping an eye on them, finding and categorizing intrusions are some of the methods put in place to enhance cloud security. As a result, one of the most important lines of protection against assaults on the cloud computing network performs the function of an IDS. What present intrusion detection technologies may do to accurately identify cyberattacks is compromised due to their inability in order to manage and evaluate the enormous volumes of traffic present in cloud environments all at once. In order to effectively identify cyberattacks in the cloud environment, this study suggests a model constructed using deep learning. The foundation of the model's structure is sophisticated convolutional neural networks (CNNs). Collecting datasets, preprocessing them, selecting features, training the model, testing it, and finally evaluating its performance are all crucial steps in the suggested CNN-based intrusion detection technique. The suggested paradigm successfully defends cloud networks against a wide range of assaults, as shown in the experiments. In terms of detecting and classifying network intrusions, the model has shown its worth with an accuracy, precision, and recall of more than 98.67%. Extensive testing has shown that the approach successfully safeguards cloud security protocols while reducing vulnerability to new and emerging security concerns.

[24] Discussed the growing role of Machine learning and deep learning are two examples of AI-based approaches that have brought data mining and information extraction from data into the spotlight in recent years. An instrument that can provide testing sets that can accurately identify SQL attacks. A previous study of how security audit tools operate will serve as the basis for the tool, namely those that are dedicated to SQL pen testing. The data used in the investigation came from web server logs that were created in a pen testing lab. Every potential entry point has been categorized according to the amount of risk it poses, using the shared patterns derived from the logs. As a last step, a classifier system was trained with the generated data to obtain a high performance in detecting attacks ranging from 97% to 99%. By sharing the training data with other servers, we can build a distributed network that can detect attacks and legitimate petitions, and then notify all linked clients in order to halt the IP petitions address of the attacker.

[23] Highlighted that the number of assaults has grown in tandem with both the birth of the Internet and the subsequent explosion in its user base. Protecting data on computers and networks is a challenging endeavor. In order to safeguard computer and network systems against intrusions, for the purpose of detecting intrusions. This study set out to compare the

efficacy of four distinct approaches to detecting network attacks: Support Vector Machines, Decision Trees, Gaussian Naive Bayes, and Random Forests. The performance of these classifiers was compared using a dataset that was recently updated. In terms of assault detection, the experimental findings show that both Decision Tree and Random Forests worked well.

[22] discussed about great performance, adaptability, and scalability, NoSQL databases like Cassandra and MongoDB have seen fast adoption in the last several years. When contrasted with SQL databases, these databases raise additional security issues. There is a higher risk of fraud, invasions, and data breaches in NoSQL databases due to their availability-focused design, lack of control over access, and changeable schemas. In this work, we take a look at how NoSQL databases may benefit from sophisticated machine learning algorithms to detect fraud and breaches more effectively. We look at several different machine-learning methods, including support vector machines and neural networks. In order to detect suspicious patterns of access that might indicate malicious activity, Clustering, random forests, and random forests are all useful tools for analysing the activity logs of large datasets. We take a look at the online training of these models to identify new risks, and we test the methods using a prototype NoSQL database to make sure they work. With minimal false-positive rates, our findings demonstrate excellent accuracy in identifying injection attacks, illegal queries, and irregular database activity.

[21] emphasized that to finding a reliable method to detect intrusions on networks is the fundamental objective of those studying intrusion detection. An important strategy is to monitor network activity in order to detect attacks. Intrusion detection systems (IDS) are security measures intended to safeguard network infrastructure and data from unknown threats. The design of these systems has been greatly influenced by machine learning (ML) methodologies communication from network assaults. Nevertheless, ML approaches' performance is impacted by reliance on current communication technologies and the consequent growth in network threats. We go over the fundamentals this article discusses intrusion detection using machine learning. We go over the processes that ML methods used to identify and categorize breaches. To top it all off, our study lays out the current cutting edge of intrusion detection using machine learning and classification, including all the pros and cons of each. The study also provides a synopsis of previous studies that have focused on ML-based IDS. We want to cover a lot of ground in this article on the problems

with ML-based IDS. We go on to talk about potential avenues for future study that might improve the efficacy and efficiency of IDS. If you are a beginner researcher interested in ML-based IDS, this review will give you a leg up.

[20] investigated the role of safety, dependability, and cybersecurity of autonomous cars may be greatly improved using the most up-to-date intrusion detection systems (IDSs), the subject of this research. Using ensemble learning and K-nearest neighbours, two non-tree based machine learning approaches, this study proposes a way for creating an intrusion detection system (IDS) tailored to autonomous automobiles. The scalability, real-time adaptability, and reduced likelihood of overfitting made these algorithms ideal for processing complicated and massive information. By efficiently managing skewed data, these algorithms improve the detection accuracy of both benign and malicious occurrences. The suggestion for an intrusion detection system that does not rely on trees (NTB-MTH-IDS) was evaluated using the usual evaluation criteria, such as recall, specificity, and sensitivity, as well as accuracy and precision. The evaluation was conducted on the CAN incursion, CICIDS2017, and NSL-KDD datasets are all derived from real-world sources. The results demonstrate that K-nearest neighbors and stacking, components of NTB-MTH-IDS, reach 99.00% accuracy in the CICIDS2017 dataset, 98.57% in the NSL-KDD dataset, and 97.57% in the CAN dataset, respectively. Further, the F1-scores are as follows: 99.00%, 98.79%, and 97.54%. Guaranteeing the reliability and security of autonomous cars may be achieved by the outcomes of this study, which can lead to the establishment of a strong intrusion detection system. Motorists, passengers, and pedestrians are protected from the repercussions of any cyber dangers thanks to this accomplishment.

[19] Highlighted, more people are using online apps that are powered by databases. Also on the rise are assaults on such online apps. The SQL Injection attack is a typical kind of attack against online applications. Injection of malicious code or SQL queries into an application using client-side input data is one example of this kind of attack. Despite the abundance of detection methods, most of them have concentrated on application-level SQL structures. Therefore, some of the database-level assaults went undetected by those methods. For database-level attack detection, current methods rely on categorization algorithms and appropriate kernel functions. Since SVM classification is an approach to supervised learning, and it is unable to identify unknown threats. This research presents a hybrid framework that combines the semi-supervised Efficient Data Adaptive Decision

Tree (EDADT) approach using supervised support vector machine (SVM) classification algorithm. It uses the database log-derived internal query tree for best framework performance. A multi-dimensional sequence is used to convert the query tree into an n-dimensional feature vector, and then the internal query tree is obtained. One part of the feature vector is the semantic features. The generation of multi-dimensional sequences is further assisted by the syntactic and semantic features. If the feature has a string value, it is transformed to a numeric value after extraction. The proposed approach demonstrates superior performance compared to the current ones when it comes to assault detection accuracy.

As data volumes continue to grow at an exponential rate, big data analytics has emerged as a crucial strategy for several companies. The data that has been gathered may include crucial information that might be the answer to many issues in fields including healthcare, marketing, cybersecurity, intrusion detection, and fraud. An intrusion detection system (IDS) monitors and analyses data in order to detect any possible entry into a system or network. Due to the massive volume, variety, and speed of data provided via the network, the data analysis process is now prepared to detect any attacks using normal ways. These approaches face tremendous hurdles. The efficiency and accuracy of intrusion detection systems are enhanced with the use of big data analytics. Eight machine learning techniques used by intrusion detection systems (IDS) are described in detail here as well as intrusion detection techniques, their pros and cons, the various types of IDS, the pros and cons of each type of IDS, and five datasets utilized by IDS. Through their comprehensive review, the authors also aim to provide a solid image of the methods and information gathered for the purpose of intrusion detection systems [18].

Everyone in an information-rich group is capable of making mistakes; some are more dedicated to discovering solutions to database security issues, while others are more willing to take risks for unknown reasons. Database security research aims to safeguard databases against unauthorized access, deletion, or alteration. The main works on database security research over the last several years are summarized in this article. We begin by categorizing these articles according to the characteristics that have an impact on database security. Some concept explanations are interleaved to make these techniques simpler to grasp, in comparison to conventional and machine learning (ML) approaches. As a second point, although the relevant study has produced some promising outcomes, it has also had several

drawbacks, such as a lack of strong generalization and a departure from reality. Then, we suggest several avenues for further investigation in this area. We conclude by reviewing the key takeaway [102].

[16] The exponential surge in data output over the last decade—anywhere from a zettabyte to a petabyte—has led to massive expansion networked computers and IoT devices throughout the globe. The expansion of the network has also brought forth new security concerns. But it's hard to do intrusion detection with massive data like this. Modern networks have many potential advanced uses, including urban areas, smart grids, appliances, and everyday items, online banking, e-government, and e-commerce. Due to the prevalence of security and privacy concerns in computer networks, various IDSs have been created in recent times. In the event that intrusion detection system protection fails, data availability, integrity, and confidentiality will be compromised. To deal with these sophisticated threats, traditional methods just do not cut it. There have been suggestions for advanced deep learning techniques that are capable to automatically detect network intrusions and identify anomalous activity. An all-encompassing Studying intrusion detection system via the use of deep learning techniques is the objective of this study. This study takes a close look at and analyses publicly accessible network-based IDS datasets. Critiques of deep learning algorithms for IDSs have focused on their recall, accuracy, precision, f-1 score, false alarm rate, and detection rate, among other performance metrics. Security and privacy in networks have also been addressed, along with the problems that currently exist and potential remedies.

[15] When it comes to protecting networks against sophisticated assaults, an IDS is an essential piece of equipment. There have been several proposals for ML and DL algorithms with the aim of building anomaly-based intrusion detection systems (AIDS). Some problems with related work have been identified by our literature study on AIDS. These include using outdated datasets, doing superficial analyses and validation of the findings, and using methods, parameters, and testing criteria that are randomly chosen. Applying a set of standards to different datasets and attack types, this study thoroughly evaluates prior research on AIDS. The goal is to provide benchmarking results that may uncover the most appropriate algorithms, parameters, and testing criteria for AIDS. In order to find efficient and effective This research employs eleven well-known supervised and unsupervised ML approaches to address the ML-AIDS of computers and networks. Supervised ML

algorithms include things like artificial neural networks (ANNs), whereas unsupervised ML algorithms include things like k-means, self-organizing maps (SOMs), and expectation-maximization (EM). Both types of algorithms perform machine learning tasks. An ideal classifier assessment is attained by introducing many models of these algorithms and then investigating their respective turning and training parameters. This research takes a different approach than others by assessing the efficacy of AIDS using metrics like as recall, 31 ML-AIDS models were evaluated for accuracy, precision, and F-Score. Taking into account the significance of temporal complexity in AIDSs, the effectiveness of ML-AIDS models is assessed by consolidating their training and testing times into account. To put the ML-AIDS models to the test, we use the CICIDS2017 dataset, which includes real-world network assaults and is relatively new and very imbalanced across classes. When compared to other models that provide inconsistent and subpar results, models known as k-NN-AIDS, DT-AIDS, and NB-AIDS tend to perform better and identify online attacks more effectively.

SQL Injection is still among the worst security flaws, exposing users' private data and causing financial losses. The most current OWASP Top 10 study ranks injection attacks as the top vulnerability, and the frequency of these attacks is on the rise. Intrusion Detection System (IDS) rules that work based on static signatures include a common component of traditional defensive systems [14]. These rules are great for preventing known attacks but aren't effective against zero-day threats. Machine learning approaches are now the focus of a lot of study. These methods may identify undiscovered assaults, but their performance costs might add up depending on the methodology. To add insult to injury, some intrusion detection systems scour database server logs for information, while others focus on gathering traffic entering the web application from the web application host or the network. Here, we're gathering data from two places: the server hosting the web app, with a Datiphy appliance node in the middle, and the MySQL database server on each side. We found that by analyzing these two datasets together with a third, correlated dataset, we can achieve performance levels that are comparable to, if not better than, those of neural network algorithms when it comes to accuracy. This was achieved using algorithms like rule-based and decision trees.

By linking inanimate devices to the web, the proliferation the IoT has changed the way we live our lives in many ways by establishing automated, interactive settings. Nevertheless,

there are significant security problems that arise from this fast development, especially with relation to intrusion detection. One of the defining features of the IoT is its dynamic and diverse networks, which are often incompatible with traditional intrusion detection systems (IDSs). Thanks to its intelligence and adaptability, machine learning is showing great promise as a solution to these problems, allowing us to tackle complex and ever-changing threats. In this extensive research, For the goal of intrusion detection in IoT systems, many machine learning approaches are investigated, such as supervised, unsupervised, deep learning, and hybrid models. It exemplifies how machine learning may improve the safety of IoT networks by assessing their efficacy, limits, and practical applications. The report also looks at what's happening in the business right now, which is great since the IoT security ecosystem is changing so fast, so there has to be constant research to stay up [13].

[12] Since many companies rely on databases to hold vital company data, these repositories are often the focus of security breaches. The vast majority of attacks against websites that provide dynamic content use SQL Injection. To lessen the impact, companies include Intrusion Detection Systems (IDS) into their security architecture to identify and prevent such attacks. While there is a thorough cutting edge in identifying SQL Injection assaults, the authors note that there is a gap between the state-of-practice for current tools that can identify these attacks. A few exceptions exist, such as Bro and Mod Security, which provide some protection against SQL Injection attacks, but overall, most intrusion detection system implementations do not. By comparing these tools on the CSIC dataset, the authors investigate current methods for protecting databases against SQL Injection attacks and determine what features and implementation details intrusion detection systems (IDSs) require to be effective. The results show that when it comes to protecting against SQL Injection, signature-based IDS is the way to go.

[11] Computer and network security nowadays, it's one of the biggest problems. Intrusion Detection Systems are one way to keep data stored in databases secure. Identifying fraudulent transactions that damage data is the main goal of intrusion detection technologies implemented into database administration platforms. Data mining algorithms are now being developed by academics to identify these types of fraudulent database transactions. Their method is centered on mining data dependencies among data pieces. The deals that fail to adhere to these data requirements, however, are labeled as malicious. These methods' data dependency miner designs are based on algorithms that have certain

restrictions. As an example, they may encounter weak data dependencies when conducting experiments to find the optimal values for minimal support and associated restrictions. One new data mining technique is ODADRM, or Optimal Data Access Dependency Rule Processing approach that we provide in this research as a means of developing an intelligence data system (IDS) data dependency miner. One improvement over the original k-optimal rule discovery algorithm is the ODADRM technique tailored to the database intrusion detection space. In contrast to earlier data dependency miner techniques, ODADRM does not suffer from the same drawbacks. The end conclusion is that our method outperforms the competition when it comes to tracking legitimate transactions and identifying fraudulent ones.

[10] It has long been an objective of cybersecurity, particularly intrusion detection, to identify zero-day breaches. Many models have been suggested as potential solutions using machine learning, but a workable one has not materialized yet, mostly because of the limitations imposed by the availability of outdated free datasets. In this research, we provide a Zero-Day intrusion detection method that uses machine learning to train on data generated by CICFlowMeter's flow-based statistics. In light of the accuracy, recall, F1 value, AUC, and time overhead outcomes of their cross validation, the eight most popular machine learning classification models are chosen. At last, the suggested system is evaluated using the testing dataset. The testing datasets are intentionally constructed to include new forms of incursions that were not learned during the training phase. This allows us in order to evaluate the studied models' efficacy and feasibility. The dataset’s usual data comes from actual traffic flows that are created from everyday usage. With an acceptable amount of time overhead and an accuracy of almost a hundred percent accuracy with close to zero false positives, encouraging findings have been achieved. We contend that some machine learning models—including the MLP classifier, the K-Neighbor classifier, and quadratic discriminant analysis—are capable of identifying zero-day assaults with satisfactory performance when fed correctly with flow-based statistical data.

[9] The need for trustworthy and secure internet has increased in popularity due to the availability of high-speed Internet has grown. Both the sophistication and the intensity of network assaults have been on the rise lately. Consequently, an increasing number of businesses are opening themselves up to potential attacks. In order to achieve a greater speed of detection while simultaneously decreasing the number of false alarms, this study

aims to categorize network threats using neural networks (NN). Two forms of categorization are discussed in this paper: one with a single class (normal or attack) and another with many classes (normal, DoS, PRB, R2L, U2R), with the NN additionally detecting the attack category. The architecture's complexity, symbolic data translation, and training data splitting are all evaluated by extensive study. In this research, we take a look at two engines: BPNNIDS, which stands for two acronyms for "radial basis function neural network intrusion detection system" and "back-propagation neural network intrusion detection system," respectively. This study presents two engines that are tested on the same dataset: the International Knowledge Discovery and Data Mining Tools' DARPA 98 KDD99 benchmark dataset. The engines are compared to both conventional and alternative machine learning techniques. In comparison to other approaches described in the literature, BPNNIDS demonstrates a considerable improvement in reaction time, rates of detection and false positives.

[8] The rapid digitization of corporate processes and increased familiarity with the digital world were both prompted by the coronavirus epidemic, which affected many firms all throughout the world. A large number of people holding positions in IT, such as web developers and system engineers, are therefore subject to various dangers about which they may not be fully informed. Injection of harmful data into a database via an API call request using SQL syntax is one example of a vulnerability that attackers may exploit. Services for networks and infrastructures provide businesses several benefits. On the other hand, the same benefits could leave the company open to certain threats. Here, we'll examine SQL injection threats in detail and compare a number of machine learning techniques "Naive Bayes" and "K-NN," support vector networks (SVN), decision trees, and a deep learning algorithm that are able to detect these kinds of assaults. According to CNN, this comparison will demonstrate the possibilities of cloud security algorithms that use machine learning.

[7] Introduced Intruders are constantly targeting the network area, and their attacks are getting more spontaneous and difficult to stop, making network security a major worry. There is a need for further development in this area because numerous sophisticated attacks can bypass endanger security aspects including availability, confidentiality, and integrity, and bypass most network security mechanisms despite the fact that many models, pieces of software, and hardware have demonstrated promising results in detection and incident response rates. This study developed and deployed a Model for an Intrusion Detection

System that makes use of CNN features harvested from the CIC-IDS 2017 dataset. Examining the many Attack kinds in the dataset allowed for feature learning and modeling to be accomplished. Python was used to develop the suggested system, and recall, precision, and accuracy are the metrics used to quantify performance. In comparison to other models, the findings obtained the Convolutional Neural Network model's approach to Intrusion Detection fared better, with an accuracy of 99.78%, as shown by analyzing the accuracy of the CNN-IDS model's predictions.

[6] Emphasized this work neatly categorizes the current literature on intrusion detection systems (IDSs) that employ deep learning (DL) methodologies, aiming to fill a gap in this vital area of research. Xplore, which is part of the Institute of Electrical and Electronics Engineers, is one of four primary databases that include articles. The other two are Scopus and Web of Science—that include the terms "deep learning," "intrusion," and "attack" or variants thereof are the primary emphasis. All of the technical literature may be found in these databases. The collection includes sixty-eight articles. Articles that provide a method for assessing or recognizing intrusion detection strategies via the use of DL account for the lion's share (72.06%; 49/68). Studying and applying publications to the DL field, IDSs, or related concerns accounts for the second greatest share, at 22.06% (15/68). Fourth most common (5.88 percent; 4 out of 68) topics are paradigms and frameworks for implementing or operating IDSs. The writers' suggestions, extensive study, open difficulties that limit the technology's applicability, and the field's fundamental features are derived from these sources. Following that, we will go over a mapping of results for potential future paths. In order to satisfy the requirements of accurately detecting established a three-stage methodology for distributed denial-of-service assaults. For academics curious in DL-based IDSs, this paper offers a wealth of background resources.

[5] Explained these days, databases are an essential component of every online business or e-commerce platform. No matter what technology is utilized, these databases are susceptible to SQL injection attacks. Both attackers and intruders find these attacks to be very hazardous and straightforward to employ. A novel method for detecting SQL injection intrusions is detailed in this research. We propose this strategy in light of the notion of creating trustworthy user profiles from authorized users' Queries utilizing association rules. Following that, we will use a data-mining-based hybrid (anomaly + misuse) detection model to identify queries that depart from our typical behavior profile. An XML version of

the typical behavior profile will be generated. We can reduce the number of false positives in this manner.

[4] Protecting computer networks against the importance of cyber-attacks is growing as the number of applications that rely on them is growing at an exponential pace. Consequently, intrusion detection systems (IDS) that can spot oddities and hostile actions in networks are in high demand. An effective data-driven intrusion detection model may be built using science and technology of artificial intelligence, with a focus on machine learning techniques. In order to detect breaches and offer intelligent cybersecurity services, we employed a number of popular machine learning categorization methods in this article. Some examples of these methods are decision trees, decision forests, artificial neural networks, decision tables, and naive Bayes classifiers. Finally, we measure the effectiveness using fscore, precision, accuracy, and recall by doing tests on cybersecurity datasets that include various types of cyber-attacks.

[3] There has been a plethora of study on intrusion detection systems (IDS). Achieving a consistent detection rate has become a significant problem for intrusion detection systems (IDS) due to the ever-changing nature of cyber threats. In spite of several machine learning techniques aimed at making IDSs more efficient, current intrusion detection methods are still not very effective. The dataset used in this work is the 2017 CICIDS Information about to intrusion detection systems compiled by the Canadian Institute for Cybersecurity. on deal with missing data and address the problem, we applied multiple imputation by chained equations (MICE) on the dataset multi-class unbalanced classification issue. Additionally, an estimator of decision tree classifier was used in conjunction with the recursive feature elimination (RFE) approach to decrease the feature count by computing the relevance of each feature. To enhance the identification of minority classes, the synthetic minority oversampling method, in conjunction with the edited nearest neighbor (SMOTE-ENN) combination, was used to resample the training data. Four machine learning methods—K-nearest neighbor, XGBoost, LightGBM, and random forest—were taught and assessed in this study. To facilitate further experimentation with hyperparameter adjustment, Optimizing Bayesian Models with Tree-structured Parzen Estimators (BO-TPE) was used to examine the relevance of each model's hyperparameters. Every one of the four ML methods reached a minimum of 98% on the three performance measures: accuracy, MCC and AUROC are the areas under the receiver operating characteristic curve.

To communicate with a database, one uses SQL, a database language. Database creation, modification, and deletion are all made possible with the aid of SQL. Almost every company these days has its own database, and some of those databases may include sensitive information that shouldn't be made public. Nowadays, SQL injection is among the most prevalent forms of online assault. Identifying and fixing SQL injections, the presentation's primary focus is on SQL injection attacks and related issues. Using a machine learning approach called clustering, this study presents an algorithm that can identify SQL injection attacks and unauthorized users by keeping an audit record [2].

Extremely consistent health solutions are required due to the emerging pace of network security threats. To identify incursions, researchers often operated on many fronts at once. Using a machine learning technique, this study discusses the security aspects of intrusion detection. Meanwhile, intrusion detection systems (IDSs) were vital in developing robust connection frames, which essentially safeguard systems by identifying and intercepting a multitude of attacks [1]. There are a lot of new approaches that rely on machine learning techniques. However, they may not provide 100% accuracy in identifying all types of violations. We provide a thorough evaluation of many ML methods for identifying the root causes of problems with these methods when it comes to detecting intrusive actions. We also go over the limitations that come for every one of them. Machine learning data mining technologies abound included in the article as well. When evaluating and estimating the implementation of a detection framework, consistent reference datasets are crucial. To measure the effectiveness of intrusion detection methods, numerous datasets are employed; case in point: DARPA98, KDD99, ISC2012, ADFA13, and so forth. However, in our study, we used the most recent one, CICIDS2017, which yielded much improved accuracy. As part of this research, we developed an estimation methodology for IDS datasets and began a thorough assessment of the current datasets using our own anticipated criteria. We maintain this advantage by identifying problems dealing with nat intrusion detection and offering a framework for used for anomaly detection research in the future.

This study provides an explosion in the popularity of the variety and velocity of network threats. Network security technologies include a wide variety of methods, including firewalls, antivirus, malware, and spam filters. To identify suspicious or illegal activity on a network, A powerful and effective technique for network security is an intrusion detection system (IDS) [77]. We review the present status of NIDS research in this paper, the methods

used to study it, and the datasets that are most often used to test IDS models. This paper presents an analysis considers the following factors: annual publication volume of intrusion detection articles, average citation count per article, and top cited research papers on intrusion detection systems in journals and conferences. Based on publications in the intrusion detection sector throughout the previous fifteen years, this article discusses NIDS state-of-the-arts, widely used NIDS, citation-based analysis of benchmark datasets, and NIDS approaches for intrusion detection. Research in this area also provides a comparative analysis based on citations and publications. This study serves as important resource for beginners and researchers interested in Network Intrusion Detection System (NIDS).

[78] Pointed out most manufacturers neglect security requirements during design, network security becomes the primary concern in IoT networks. Firewall, intrusion prevention, and intrusion detection system effectiveness is accuracy dependent, hence reducing false alarms requires an improvement in detection rate. A firewall's principal function is to filter incoming data packets for malicious content and block them according to established policies. But with modern attack methods, it's almost impossible to tell bad traffic apart from normal. A two-pronged hybrid strategy is suggested in this research to solve these problems. To start, in order to make the suggested framework more accurate, the genetic algorithm (GA) is used to pick out relevant traits. The next step is to use the popular ML method, which may include the ensemble classifier, decision tree, and support vector machine (SVM). Through the utilization of a multiclass NSL-KDD database and a 10-fold cross-validation technique, a 99.8 percent accuracy was achieved.

This study provides a way to enhance the effectiveness and efficiency of Network Intrusion Detection Systems (NIDS) by using Machine Learning (ML) techniques, including Decision Trees (DT), Naïve Bayes (NB), and Support Vector Machine (SVM)[79]. A thorough assessment and comparison of these algorithms is part of the suggested technique. Next, the UNSW-NB15 and NSL-KDD datasets will be used along with common evaluation measures including F1-score, accuracy, and precision. The research finds the best algorithm for NIDS deployment in the real world. Strengthening network security against emerging cyber threats is a continual endeavor, and our study greatly helps by offering practical insights and suggestions for deploying the best suited ML algorithm.

[80] Highlighted that different types of attacks may compromise wireless networks security risks due to their vital role in current communication systems. In addition to identifying

and reducing these dangers, intrusion detection systems (IDSs) may effectively scan network traffic for malicious actions intended to exploit the network's categorization, honesty, reality, and accessibility. In order to detect and categorize network assaults, the approaches of deep learning (DL) and machine learning (ML) work well. Using ML and DL models, this study introduces a novel wireless intrusion detection system that can detect and classify different kinds of network attacks. Improved IDS detection efficiency and accuracy, as well as IDS scalability and wireless network performance estimation, were all outcomes of the suggested approach. Additionally, it trains many IDS models with respect to suited wireless networks, studies IDS strategies employ machine learning for the development and deployment of intrusion detection systems in wireless networks. It compares and contrasts the display of current processes with those of suggested models. Thus, the proposed system may be used to effectively identify and categorize network threats as they occur in real-time, serving as an intrusion detection system (IDS) for wireless networks.

[81] emphasized that its core, the capacity of endpoint devices to collect data is crucial to the IoT massive amounts of data being produced and transmit it to a server. The early detection of these harmful behaviors helps protect data from assaults on IoT networks, which are susceptible to them. In order to identify potential dangers in IoT networks, this research mainly aims to create machine learning models. Building a model requires both normal and attack data generated by the IoT environment. Using a wireless network, DHT11 sensor, and Node MCU ESP8266, a test bed is constructed to mimic an Internet of Things (IoT) environment. A laptop computer is used to construct an adversarial system that may launch poisoning and sniffing assaults. Sensor data, including humidity, temperature, and due-point, is sent to the Think Speak platform over a wireless gateway. During the normal phase, the Node MCU stores and labels as normal data the sensor values that it transmits to the Think Speak server. Though the Think Speak server and Node MCU are exchanging data, an adversary system may covertly intercept and alter this data during the attacking phase. Data collected during an ARP Poisoning A network intrusion known as a "Man in the Middle" assault attack data during the attacking phase. Data may be classified into normal and attack classes using machine learning algorithms such as Adaboost, Naïve Bayes, SVM, decision trees.

[82] Highlighted that there are a lot of various sorts of assaults that are putting data on the Internet at danger right now. Data security on the network is, hence, an important issue. The purpose of developing the Intrusion Detection System (IDS) was to notify the network system administrator in charge of network security whenever it detects the beginning of a series of assaults, stopping the loss of very important data. Firewalls are an extrapolative paradigm capable of differentiating between legitimate and malicious network traffic. A paradigm shift in software-driven networks has occurred with the advent About SDN, which divides networks into a control plane and a data plane. With software-defined networking (SDN), the data plane and control plane are separated paves the way for a programmable and managed network, where programs running on the top plane may communicate with physical devices via the controller. Managers are in charge of running the various network modules on the control plane and creating flow rules that direct packets to the data plane switches. Hackers aim their attacks at the software-defined networking (SDN) controller in an effort to take over the control plane, symbolizing the network's central nervous system. The level of control provides a lot of useful functions, like controlling flows to lower-level data plane devices through southbound APIs and implementing complex Northbound APIs allow devices in the application plane to execute business and application logic at a higher level. The control plane's concentration, however, makes it an attractive target for security threats. An extensive literature evaluation of seminal works published on the topic of intrusion detection systems (IDS) from 2015 to 2021 is presented in this research. to safeguard software-defined networking (SDN). In addition, we provide two comprehensive taxonomic research examining several intrusion detection systems (IDS) for software-defined networking (SDN) security, structured according to the learning categories of the ML-DL algorithms used. Within the SDN paradigm, intrusion detection systems are constructed using a handful of benchmark datasets have also been the subject of our cursory investigation. At the end of the study, this study addresses some of the ongoing problems with intrusion detection systems (IDS) and how they affect SDN security.

[83] Highlighted that the rapid growth in technology use, the amount of data processed across the Internet has grown substantially over the years. Sheer data security is crucial due to the massive volume of data sent online. An intrusion detection system (IDS) could be helpful in this situation as it can help find possible threats to virtual security. Intrusion detection systems (IDS) monitor data in order to detect any intrusion attempts into a system

or network. Hackers use a plethora of methods to breach security and access a network. The planned IDS is being built using state-of-the-art technologies, including machine learning techniques, with the purpose of classifying attacks, detecting them whenever they happen, and deciding which machine learning approach is most suited to identify the attack. Naive Bayes and Support Vector Machine (SVM) are two of the machine learning methods used. Classification problems are common targets for these approaches. Discoveries in NSL-KDD Analyzing intrusion detection systems require a dataset. The findings prove that Support Vector Machines outperform Naive Bayes. In order to compare and contrast different categorization methods, This study assess the accuracy and misclassification rate of Support Vector Machine and Naive Bayes.

[12] Discussed businesses rely on databases to keep their secrets safe. This highlights the vital need of protecting them against data loss or theft. Database Intrusion Detection Systems (DIDS) are often used to address database system intrusions. The article summarizes the most important popular database intrusion detection methods and addresses concerns related to using them on the database server layer. Based on the observed weaknesses, the majority of DIDS do not successfully handle various aspects of database systems. This includes, but is not limited to, issues with data warehousing alert management and ad hoc assignments. This study forms the basis for the presentation of research issues, as well as the proposal of requirements and suggestions for the development of improved or novel DIDS. One important takeaway is how difficult it is to create and compare DIDS that are a perfect fit for any given business's environment. We think this effort is a great way to get people talking about security and database research.

[94] Explored the use of Machine Learning techniques for detecting and preserving online threads. Online fraud detection and prevention using trustworthy machine learning assaults requires addressing the significant problem of unexpected results, both favorable and bad. We aimed to find out why there were false-positive and false-negative results. We ran our tests on the CSIC 2010 HTTP dataset specifically. Our experimental findings show that all machine learning algorithms evaluated performed better when using the suggested fine-tuned feature set extraction to identify and classify web-based threats.

[95] Highlighted the growing concerns about data safety have arisen in response to recent developments in communication, technologies, including cloud computing and the Internet of Things. As a result of ineffectiveness of existing security measures, cyberattacks are also

on the rise alongside these advancements. There have been many recent proposals for AI-based solutions pertaining to security, including intrusion detection systems. An efficient AI-based approach to intrusion detection systems (IDS) in IoT networks is presented in this article. To solve difficult engineering challenges, we use the latest developments in deep learning and metaheuristics (MH) algorithms. To extract useful characteristics, we suggest an approach that makes applying CNNs. Furthermore, we develop a unique method for feature selection called TSODE, which combines the operators of the transient search optimization (TSO) algorithm with a new version of the differential evolution (DE) algorithm. To improve the process of balancing the exploration and exploitation phases, the proposed TSODE uses the DE. We also use three open-source datasets—KDDCup-99, NSL-KDD, BoT-IoT, and CICIDS-2017—to test the suggested strategy, which beat a lot of earlier approaches.

[84] Discussed the Cybersecurity plays a critical part in ensuring the security of our vital information and data in this era of technology. It has been our ongoing focus for years to design a more secure system capable of predicting and preventing cyber-attacks, given the high volume of assaults that occur daily. One of the goals of this study was to lay out a strategy for avoiding SQL Injection Attacks. An example of a cyberattack is a SQL injection attack, which uses malicious SQL queries to access sensitive information stored in the back-end database and change internal data. It is not intended to display this information. A database becomes susceptible to further types of assaults after a SQL Injection Attack. All of a company's data is vulnerable to a simple assault if they aren't well protected, because most firms employ SQL-based back-end databases to store data. The main goal of this research is to find the best machine learning technique for preventing and predicting SQL Injection Attacks so that a model can be built. This report provides a synopsis of our methodology, details our experiments, and discusses the outcomes of those investigations.

[85] Discussed a plethora of reliable web services and systems available today, thanks to the explosion in popularity and sophistication of contemporary communication and network infrastructures. Cybersecurity risks, however, continue to escalate. The Intrusion Detection System (IDS) is the security measure that follows a firewall in the chain of command crucial in keeping communication networks safe. When the IDS detects suspicious activity, it alerts the system or network administrator so that they may take preventative measures. This study makes three important contributions to show that various

IDS techniques can be implemented. Our primary contribution is an examination of the efficacy of applying traditional ML methods grounded in intrusion detection systems. An ensemble learning technique for detecting cybersecurity threats is proposed in the second contribution.

Thirdly, a hybrid feature selection approach is proposed to improve network attack detection. The innocuous and the most sophisticated frequent attacks are both included in the newly made available CICIDS2017 dataset., was used to test and compare all of the offered methods. For the purpose of studying their classification performance under assault across several metrics, this research takes into account various machine learning classifiers and feature selection strategies. The process of creating a reliable network intrusion detection system (IDS) includes possible with the selected minimal characteristics, according to the empirical findings of the three systems that were put into action. In addition to enhancing classification accuracy, these models also decrease the false alarm rate when it comes to classifying IDS assaults.

[86] Highlighted the serious risk that malicious user requests might compromise other user accounts, leaving them vulnerable to theft or extortion, and that these devices in the backend could be compromised as well. Sanitizing these requests before the servers process them is crucial since even a single fraudulent request may trigger a catastrophe. According to some authors, sanitizing models that use support vector machines can reliably distinguish between malicious and benign queries. With the use of a powerful classification tool, such as a decision tree, our research has shown that ensemble learning yields superior results.

[87] Emphasized the need to be effective network traffic monitoring due to the dramatic increase in internet traffic and network assaults caused by the widespread use of computers and smart devices. There have been a lot of effort to fix this, but faster answers are required. Our study shows that our distributed we built our own intrusion detection system (IDS) that uses machine learning to keep an eye on the network in what amounts to near-real-time. Similar to firewalls and antivirus software, intrusion detection systems (IDS) primarily aim to identify and prevent unauthorized access., is to safeguard communication networks. Apache Spark, the Hadoop Distributed File System, and Netmap, a tool for recording line rate packets, are used to create a secure real-time network. You may think of the IDS as having two main sections. 2) a classifier module that use machine learning as their foundation and 1) a module that processes packets. Feature extraction methods are a key

part of packet processing; we have shown their efficacy with TCP data and built a traffic classifier using Spark MLlib algorithms for machine learning. To ensure that the IDS can keep detecting assaults over a lengthy duration, the classifier routinely changes the anomaly detection rule. To find the most accurate machine learning classifier, we ran many tests. The findings and insights for further research are presented. Our intrusion detection system (IDS) may be implemented in heavily trafficked institutions and organizations with a sufficient number of nodes.

[88] Discussed that Cybersecurity is an essential area of study since networks are becoming more pervasive in our daily lives. Antivirus, firewall, and intrusion detection system (IDS) software is the mainstay of cybersecurity methods. These methods shield networks against intruders both inside and outside the network. The three essays that make up this study are... It draws attention to and enhances machine learning's uses in cybersecurity. Conventional protection measures against cyberattacks are becoming more ineffective because to the rising Internet traffic has increased, which has led to larger feature sizes and more frequent cyber incident observations. Regardless, machine learning's ability to identify and thwart cyber-attacks in their tracks is progressively becoming better. There has been an attempt to better remove hazards during the past 10 years by combining machine learning with cybersecurity. Disagreements arise while deploying data-driven intelligent systems when cyber domain knowledge and the implementation of machine learning methodologies are out of sync. Our team has reviewed the latest literature on the topic and compiled a list of the most prevalent problems encountered when using machine learning algorithms to cybersecurity. In light of these results, we have been investigating new avenues for research and experimentation with the goal of enhancing service quality and security strength.

[89] Emphasized the urgent need to enhance Intrusion Detection and Prevention systems are critically required due to the recent uptick in hacks and computer network assaults globally, which has affected companies for example, Sony Pictures (2014), Home Depot (2014), and Target (2014) etc. Network intrusions have grown in both frequency and severity. Among anomalous intrusion detection systems' most significant shortcomings is their disproportionately high false positive rates they produce. As part of our effort to enhance IDSs, this thesis employs Support Vector Machines for intrusion classification, conducts the sentence compares and contrasts several classifiers, such naïve-Bayes and multilayer perceptron’s, on the network intrusion detection classification task in order to

determine their effectiveness. In order to evaluate the classifiers, the ISCX2012 dataset is used. At 99.1 percent average detection accuracy, the proposed Support Vector Machine classifier outperforms the modified gravitational search algorithm (MGSA) neural network (97.8 percent accuracy) and the multi-objective genetic algorithm (MOGA) multilayer perceptron (97.0 percent average detection accuracy).

[90] Highlighted that the exponential growth of the internet and all the apps built for it has made it abundantly clear that the potential harm that may be inflicted by conducting assaults is growing at an alarming rate. At the same time, security systems that detect and prevent intrusions include crucial defensive instruments in the fight against complex and increasing network threats. As a result of insufficient datasets, proper there are obstacles to the implementation, evaluation, and analysis of anomaly-based approaches in intrusion detection systems. Several of these datasets, such as DARPA98, KDD99, ISC2012, and ADFA13, have been used by researchers to evaluate the effectiveness of their proposed intrusion detection and prevention strategies. Our examination of eleven datasets made available to the public since 1998 reveals that a number of these archives include information that is both outdated and inaccurate. Inadequate traffic diversity and volumes, missing feature sets and metadata, anonymised packet information and payload that does not represent current trends, and so on are some of the problems plaguing these databases. This research creates a publicly available, trustworthy dataset that includes seven typical attack network flows in addition to benign ones. The dataset also fits real-world standards. In order to determine which characteristics are most effective in identifying certain types of attacks, this study compares and contrasts a large collection employing a variety of machine learning methods to data on network traffic.

[91] Discussed the role of to find possible security breaches in databases, people utilize database intrusion detection systems (DIDS). DIDS is based on the same principles as other conventional database and network security measures, including intrusion detection and firewalls. Consequently, it is vulnerable to infiltration from both internal users and those who manage to bypass its current security measures. This indicates that, in comparison to the frequency of typical events, the occurrence of incursion events is rather unusual. Consequently, increasing the detection rate by the generation of several false alarms is a waste of resources. We want to increase the detection rate while decreasing the

concerningly high false positive rate this article presents an improved data dependence model that incorporates access control.

[92] Emphasized that the ensuring network security and accessibility is most important objective of any organization. Abnormal network access may be detected and classified using methods such as Network Intrusion Detection Systems (IDS). This means that intrusion detection systems need constant tuning to account for new types and methods of penetration. When it comes to malicious network infiltration, denial of service attacks are by far the most common. The primary objective of this study is to determine which machine learning algorithms are most suited to detect denial-of-service attacks. Three distinct machine learning techniques are evaluated using the Knowledge Discovery and Data Mining (KDD) dataset. According to the results, MLP outperforms SVM and KNN.

[93] Highlighted that recent developments in electronic computer technology have led to a dramatic increase in the storage and sharing of files and data via networks. It is more difficult for outsiders to physically access user data on a computer, as the system is walled off to them. The other side is that hackers may exploit computer networks to access user accounts and steal secret information. Preventing network assaults and ensuring data security are the main focuses of the academic community. Network traffic and host logs are the backbone of the IDS's capacity to detect and avert intrusions into the network. Still, with all of this, network managers have a lot of work to do with data analysis and rapid response strategies that are required. More sophisticated methods of protecting networks from assaults and earlier warnings of possible attacks are outcomes of AI's ability to automate hitherto manual data processing tasks. To define a standard for APT detection, we created specifically designed for APT attacks, the SCVIC-APT-2021 dataset. A Virtual Private Network (VPN) connection between two network domains is the basis of the dataset construction network environment. Using Kali Linux, an attacker may compromise two network domains from the outside and conduct several APT assaults. Each of the six APT phases included in the produced dataset employs a unique set of attack methods. The next step is to suggest using the newly created SCVICAPT-2021 dataset in conjunction in order to detect APT attacks using a machine learning model based on prior information. The macro average F1-score increases by 11.01% and reaches 81.92% when contrasted with the supervised baseline model. After that, the proposed model is evaluated against two

benchmarks, NSL-KDD and UNSW-NB15. Weighted average F1-scores of 76.42% and 79.20% are possible for both datasets.

Intrusion detection systems (IDS) may identify network attacks more effectively by using a detection technique that integrates host-based attributes with network-based characteristics. This is because devices connected to the network may retain host-based data, including system logs, after certain types of network attacks. Using the raw data from CSE-CIC-IDS2018, the SCIVC-CIDS-2021 dataset is built, which includes features based on networks and hosts. The attacking tactics are labeled on the SCVIC-CIDS-2021 to guarantee accurate categorization findings. The obtained dataset contains features with high dimensionalities; to reduce their dimensionality, we use Autoencoder (AE) for network-based features and Gated Recurrent Unit (GRU) for host-based features. Lastly, the data points are classified using PKI Difference (PKID) and knowledge-based PKI models. Averaging 96.60% on the F1-score, the PKID model outperforms the others by 7.62% compared to the results obtained just using network-based characteristics.

CHAPTER 3

RESEARCH METHODOLOGY

To determine whether incoming communication is benign or malicious, our proposed method in this study employs machine learning methods. A bespoke business chat web app running on a distant MySQL server forms the backbone of the system. There are two locations where data is recorded: first, in the HTTP traffic that flows between the servers that generate traffic and those that host the web applications. Secondly, in the data transfer between the webapp server and the offline database server using MySQL. Following processing and correlation, a new dataset is generated that incorporates information from both datasets. For our machine learning tasks, we rely on the Weka Machine Learning Framework experiments and compare different algorithms' classification accuracies and efficiency using 5-fold cross-validation throughout the periods of training and while developing the models. Our system adheres to a standard pattern, as seen in Figure 3.1.

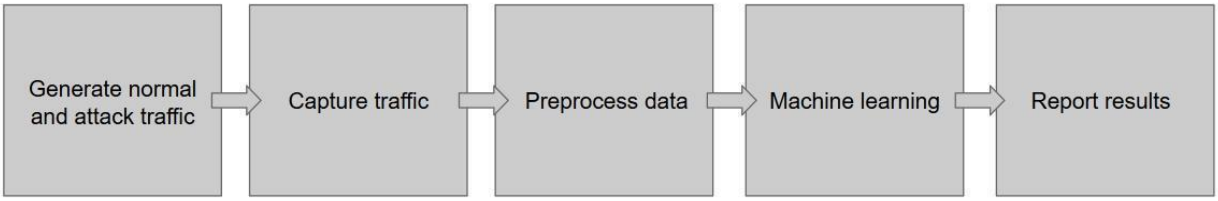


Figure 3.1: System Process

3.1 Architecture

The four server nodes that make up our design are virtual computers running on an HP server using the KVM technology that has 64G of RAM and two quad-core CPUs. Webapp, traffic generating, database, and Datiphy MySQL data capturing nodes are all part of this network. Figure 3.1 shows them, and we'll go over them quickly below.

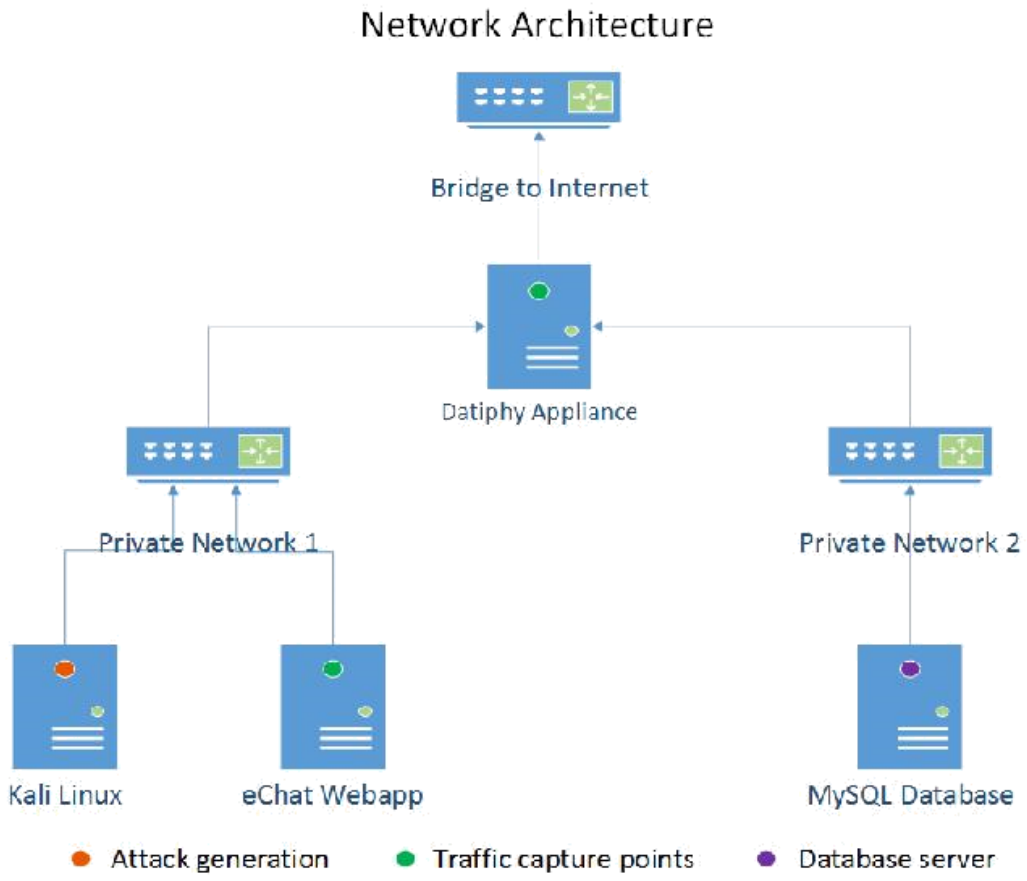


Figure 3.2: Network Architecture

Web Application Server. The webspace is where the bespoke web app is set up., and the webapp server is running Ubuntu/Apache. Located on the database server, SQL Server is the database management system that powers this website. We have two points of data acquisition, and this server is running Snort as well.

Traffic Generation Server. This machine, which runs the Kali Linux system, is used to produce both benign and malevolent traffic. Python and shell programs using the Beautiful Soup Python modules are used to produce both legitimate and malicious traffic.

Database Server. You may run Ubuntu and MySQL on this server. The webapp's chat application connects to this server remotely for database access, and these two servers are the conduits via which the webapp's MySQL traffic travels.

Datiphy MySQL Data Capture Server. This server is comprised of a virtual machine (VM) that Datiphy Inc. has made available for research purposes. For this project, we are making use of an appliance that provides insight into SQL traffic and other forms of database

activity. By routing all traffic via this Datiphy appliance, all communication View the connection between the webapp and the MySQL database server using the Datiphy web interface.

3.2 DATA PREPROCESSING

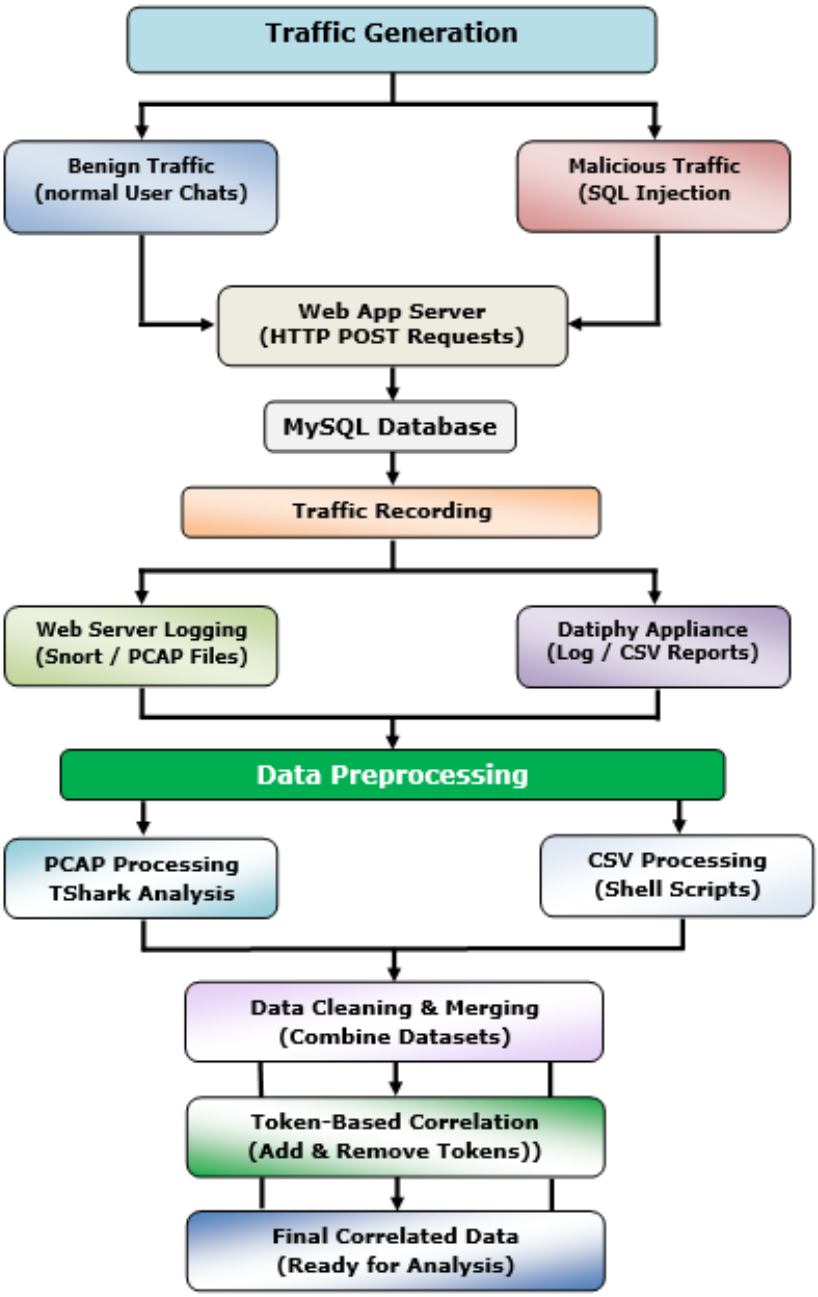


Figure 3.3: Data Preprocessing

Traffic generation, capture, and pre-processing are the three stages that make up our data creation process. Below, we will quickly go over these points. Generating Traffic. As mentioned earlier, the programs on the traffic creation server are responsible for creating we used to model both benign and malevolent traffic in our study. In this server's case, the chat webapp receives HTTP POST requests; in response, Using MySQL, the webapp server establishes a connection with the database server. The chat web app's usual traffic is made up of simulated regular user interactions. As we'll see later on, malicious traffic varies in that it includes SQL injection attacks that are explicitly programmed. In order to mimic statistically realistic online conversation, the text is created at random while maintaining a reasonable level of realism in terms of word frequency and sentence structure.

Traffic Recording. The traffic is logged in two places: the webapp server and the Datiphy appliance. We are utilizing the Snort intrusion detection system, which stores its data in PCAP files, to capture traffic where the web app is hosted. As for the Datiphy gadget, node records all of the data sent and received by the webapp and run MySQL on a distant server, and then uses the Datiphy web interface to create a report the CSV file that contains this information.

We preprocess the data. Since the webapp server stores the PCAP files generated by Snort data collection, this is also the location where most data pre-processing routines are executed. The PCAP data is processed using T Shark, which is a command line interface to Wireshark. By using T Shark, we are able to analyze the PCAP files and extract relevant data from the packets of webapp traffic. After that, shell scripts clean up the CSV file so it may be processed further. Shell scripts are used in a similar fashion to handle data collected at the Datiphy appliance. The last step is to use shell scripts to combine the two datasets into a single file, resulting in the correlated dataset. Tokens, which take the shape of distinct text strings, are introduced into the dataset in order to facilitate correlation. Once events are shown to be associated, these tokens are deleted.

3.3 WEB APPLICATION

Our approach makes use of a web app that is a programmed replica of a corporate chat app. This is hosted on an Apache webserver and built in PHP. The app's simulated chat messages

are stored in a remote MySQL database backend via an HTML form. We have intentionally left an INSERT statement open in this webapp so that SQL injection may happen.

3.4 ATTACKS

Injecting SQL functions generally, timing-based assaults, and many kinds of attacks that rely on human mistake are among the manually programmed SQL injection techniques we're utilizing for this project. Sometimes, timing-based attacks are used when the outcomes of injection attempts take a while to become apparent. Blind SQL injection is another name for these types of attacks. They often look like:

SUBSTRING (version (),1,1) CASE if it is not 4, then sleep (5); if it is, sleep (0).

An additional five seconds will be added to the request processing time when entering code like this when the first character of the MySQL version is '5'.. Due to the fact that our design uses MySQL version 5.5, this query will wait five seconds before responding, giving an attacker insight into the web application's database backend.

We are using error-based attacks, such as injecting the following code, as an example:

... or updatexml (1, concat (0x7e, (user ())),0) or ...

An XPATH problem with the database, which might include sensitive information that an attacker could exploit to their advantage (such as the username the webapp is using to access the database), is produced when this code is inserted into a SQL instruction. When working with XML documents in MySQL, you may use XPath, which stands for XMP Path Language. In these assaults, we are using two common procedures in XPATH SQL injections: UpdateXM0L() and Extract Value().

We also use attacks that insert SQL functions inside the INSERT statement, for example, user(), database(), version(), etc. To do this, a single quotation character is used to separate the statement, the function and the characters needed to complete the query are inserted, and finally, characters are used to comment out the remaining portion of the query.. Injecting the following code is an example of this kind of attack:

', (select database ()), now ()); --

The real input field will take the place of the database name when this code is inserted into an INSERT statement.

3.5 MACHINE LEARNING

We use the Weka framework, which incorporates numerous cutting-edge ML techniques, to examine the data that has been generated. Utilising the numerical and nominal data in their current forms, we use the String after loading the data from the prior pre-processing procedures into Weka to Vec filter to further process the string data into word vectors from Weka. After that, we're going to use Correlated Feature Selection to cut down on features so machine learning can be more efficient. This approach takes a collection of characteristics with low correlation between itself and one of our two classes—using the class property in this case, harmful or normal—and uses it to make a classification. Utilizing a genetic search strategy, we identify feature subsets to test for the aforementioned association using the CFS algorithm in Weka. Following the principles of natural selection, which are rooted in biology, a genetic algorithm would randomly alter a population of candidates, evaluate their fitness, and eventually develop toward a solution that is optimum. The Weka genetic search method is based on Goldberg's basic genetic algorithm.

Here are the parameters that we've put up for the genetic search: Maximum number of generations: 200, population size: 200. Those were the optimal values for our experiments. Weka starts with a mutation probability of 0.033 and a cross-over probability of 0.6. From 1611 features in the original Snort data, 57 remain after feature selection. There are a total of 56 characteristics left from the original 1656 in the Datiphy data. For the aggregated data, the characteristics are downsized from 1662 to 58. Improving experimental efficiency necessitated reducing features, which were also required in our setting for the Neural Network method to execute within realistic time constraints.

Classification in this study makes use of the following algorithms: The following have already been discussed: J48 decision tree, JRip rule-based, Random Forest, SVM, and MultiLayer Perceptron Neural Network. Developed in Java and made available as open source, J48 is a popular decision tree approach. We have a decision tree with 185 nodes and 23 levels for the correlation dataset, 97 nodes and 21 levels for the Datiphy dataset, and 101 nodes and 18 levels for the webapp dataset. JRip is an implementation of RIPPER, a rule-based approach to error reduction created by William Cohen. For the webapp dataset, our ruleset has twenty-one rules; for the Datiphy dataset, thirty-five rules; and for the correlated dataset, forty-four rules. Decision trees serve as the basis for Random Forests, an ensemble learning technique. Following an implementation by Leo Breiman, Weka's Random Forest is built upon. Weka incorporates LibSVM's support for support vector

machines (SVMs), where Knerr et al. discuss a specific variation of the SVM approach that is then put into practice. Using sigmoid nodes, Weka's MultiLayer Perceptron performs broad classifier tasks based on backpropagation. Last but not least, after much trial and error, we arrived at a Neural Network consisting of one hidden layer for each model, 29 nodes for the webapp dataset, and 30 nodes for the correlation dataset. Using 5-fold cross validation, we measured how long it took to construct the classification model and how long it took to run on the testing datasets. Below, you can see our findings for classification accuracy.

3.6 SCADA DATASET

After introducing the issue and ML methods utilized this section of the chapter aims to provide a general outline of the first dataset. This data collection originates from a CI water system that is controlled by SCADA.

3.6.1 SCADA Dataset Architecture

Two tanks, one 9 liters and the other 7 liters, make up the CI. In this example, water, each tank has the ability in order to transport or keep liquids. The four sensors in the main tank are connected to the PLC program. Figure 3.4 shows the physical architecture of the control system, whereas Figure 3.3 shows a high-level schematic description.

The main tank's liquid level may be monitored using the four sensors. The tank is low (1.25L) as "discrete sensor 0" shows. Displayed as "Discrete sensor 1" is a reading below 3.35L. According to "Discrete sensor 2," the level is 8L, however "Discrete

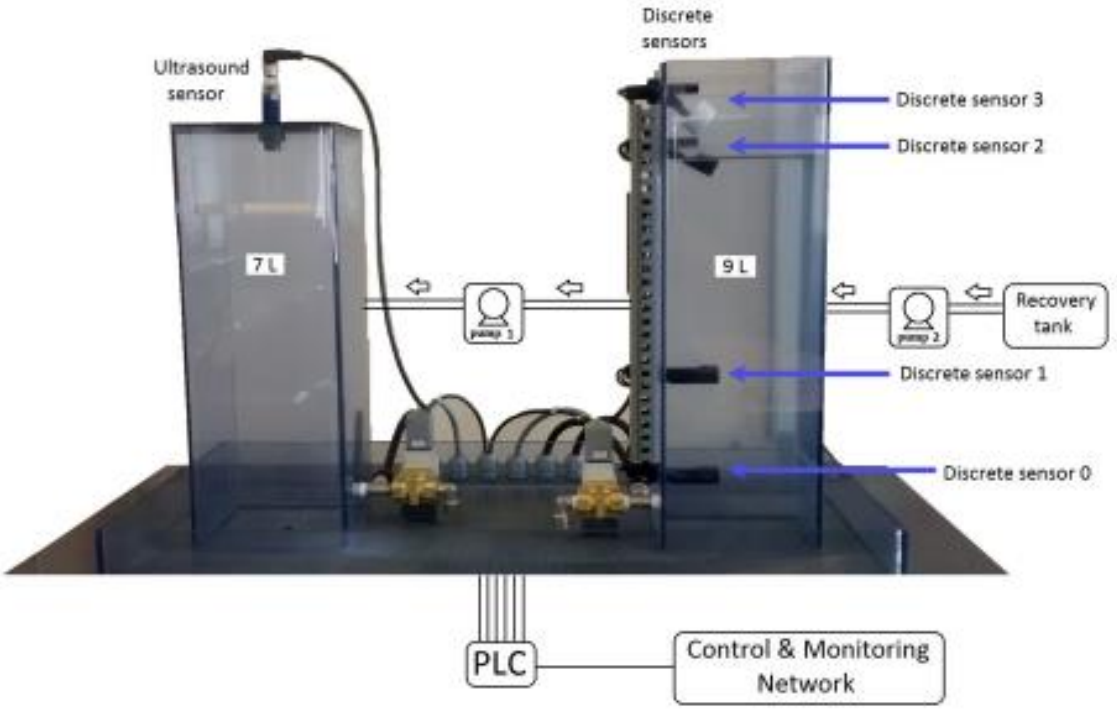


Figure 3.4 SCADA System Architecture

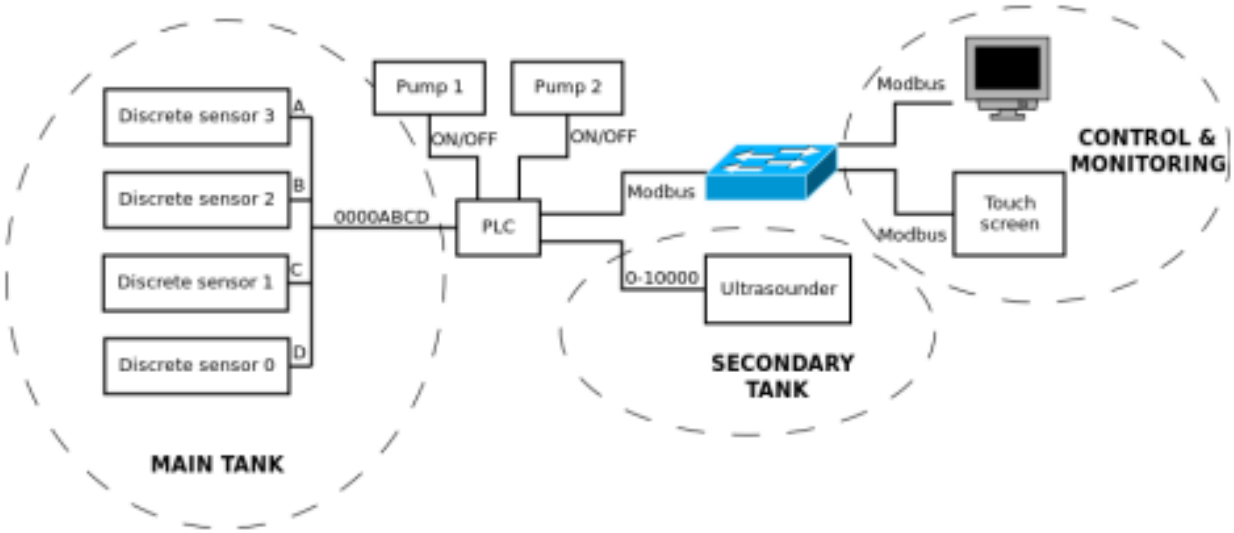


Figure 3.5 SCADA Network High-Level Architecture

The third sensor shows that the tank is full (9 liters). According to Figure 3.5, the four sensor inputs (A, B, C, and D) are combined into a single register (0000ABCD).

The flow between the primary and secondary tanks is controlled by two pumps, namely Pump1 and Pump2. An ultrasonic sensor placed on top of the secondary tank measures the amount of liquid inside. The height of the tank, relative to the liquid's surface, is used to determine the volume. You may also use it to find out whether there's any liquid in the tank. The Modbus protocol is used by all sensors in order to transmit the compiled data to the control and monitoring network.

3.6.2 SCADA Operation and Dataset Scenarios

A recovery tank mimicking a real-life liquid source—a river or gasoline line—is used to replenish the main tank during dataset production. The first tank is filled using Pump2, and to prevent any leakage, the PLC activates Pump1 to transport the liquid to the secondary tank. The valves located at the base of the tanks are used to mimic a continuous consumption of liquid.

Using sensor data readings captured at intervals of 0.1 seconds, the PLC monitors the complete operation. The PLC activates Pump 2 to enable refilling from the recovery tank if the main tank's liquid capacity falls below 1.25L.

Table 3.1 SCADA: PLC Registers Extracted Bits Representation

Reg. No.	Bit No.	Value
2	4	Discrete Sensor 3
	5	Discrete Sensor 2
	6	Discrete Sensor 1
	7	Discrete Sensor 0
3	0	Pump2
	1	Pump1
	5	Pump2 Valve
	4	Pump1 Valve
4	16-bits	Depth Sensor

Likewise, after 9L has been achieved, Pump2 will be disabled. When the liquid level drops below 2.1L, Pump1 is turned on automatically by the ultrasonic sensor. It turns off when the liquid level rises over 6.3L.

Comma-Separated Values (CSV) files including sensor readings obtained from PLC registers (2, 3, and 4) constitute the dataset. These files represent the system's status. The various registers and how they are used are summarized in Table 3.1. Each discrete sensor's binary status is provided by register 2 bits. Bitwise masks and procedures are used to get each individual sensor value. If both pumps and valves are active or inactive, it will be shown in Register 3. For the ultrasonic sensors, the depth sensor reading is stored in Register4 as a step value between 0 and 10,000. For example, a step value of 3,000 would indicate that there is 2.1 liters of liquid in the tank.

Along with typical behavior, the dataset also includes fourteen other possible outcomes. Table 3.2 details these possible outcomes, where each scenario addresses a different operating event (e.g., sabotage, accident, failure, or cyber-attack) and six impacted components. Affected components are those parts of the system that the anomaly has an immediate impact on. There are fifteen CSV files containing the dataset, one for each possible outcome.

Table 3.2 SCADA: Dataset Scenarios, Operational Scenarios, and Affected Components

	Scenario	Affected Component	Operational Scenario	No. of instances
1	Normal	None	Normal	5519
2	Plastic bag	Ultrasound Sensor	Accident/ Sabotage	10549
3	Blocked measure 1		Breakdown/ Sabotage	226
4	Blocked measure 2			144
5	Floating objects in main tank (2 objects)		Accident/ Sabotage	854
6	Floating objects in main tank (7 objects)			733
7	Humidity		Breakdown	157
8	Discrete sensor failure	Discrete sensor 1		1920
9	Discrete sensor failure	Discrete sensor 2		5701
10	DoS	Network	Cyber attack	307
11	Spoofing			10130
12	Wrong connection		Breakdown/ Sabotage	6228
13	Person hitting the tanks (low intensity)	Whole subsystem	Sabotage	347
14	Person hitting the tanks (medium intensity)			281
15	Person hitting the tanks (high intensity)			292

3.6.3 SCADA Dataset Preprocessing

The preprocessing methods that were done to the SCADA dataset are summarized in Figure 3.5. It has been shown that there are six phases to the preprocessing.

1. Case Extraction Each line in the raw CSV log files contains a reading from the dataset. Ten rows make up each instance; each row includes the time, date, number of registers, and reading or value of the PLC's registers. This step involves retrieving each scenario instance from its associated log file. For each given instance, the PLC keeps track of the values in the registers at a given moment.

2. Finding the annual percentage change for Register 4 The value of Register 4, which keeps an eye on the secondary tank's liquid level, is critical in every case. It turns out that the most important value is in Register4, but that it's not the reading value that matters, but rather the way values change over time, or the change trend or rate. For each scenario, Figure 3.7 shows the rate of change in Register.

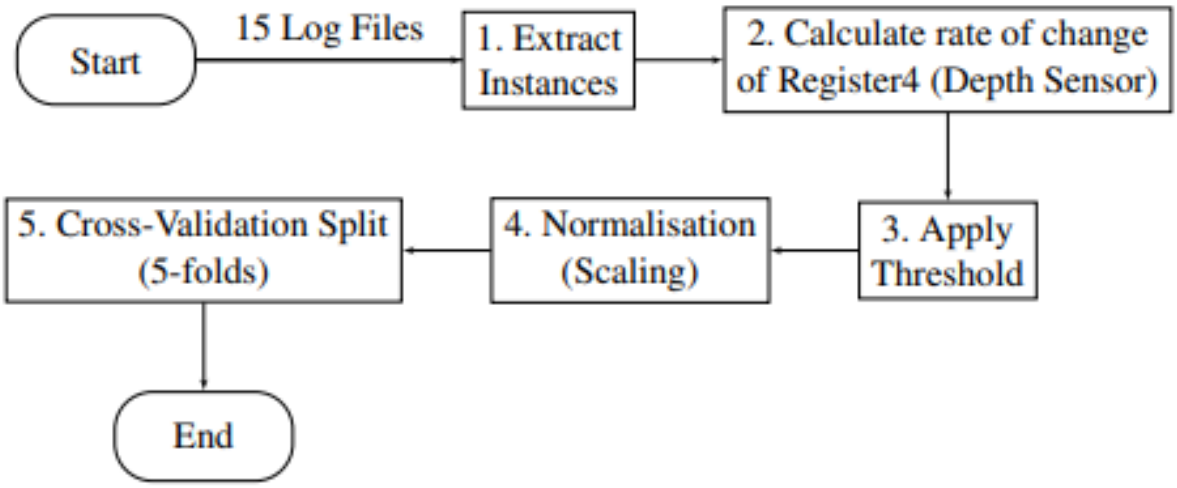


Figure 3.5 SCADA: Preprocessing Stages

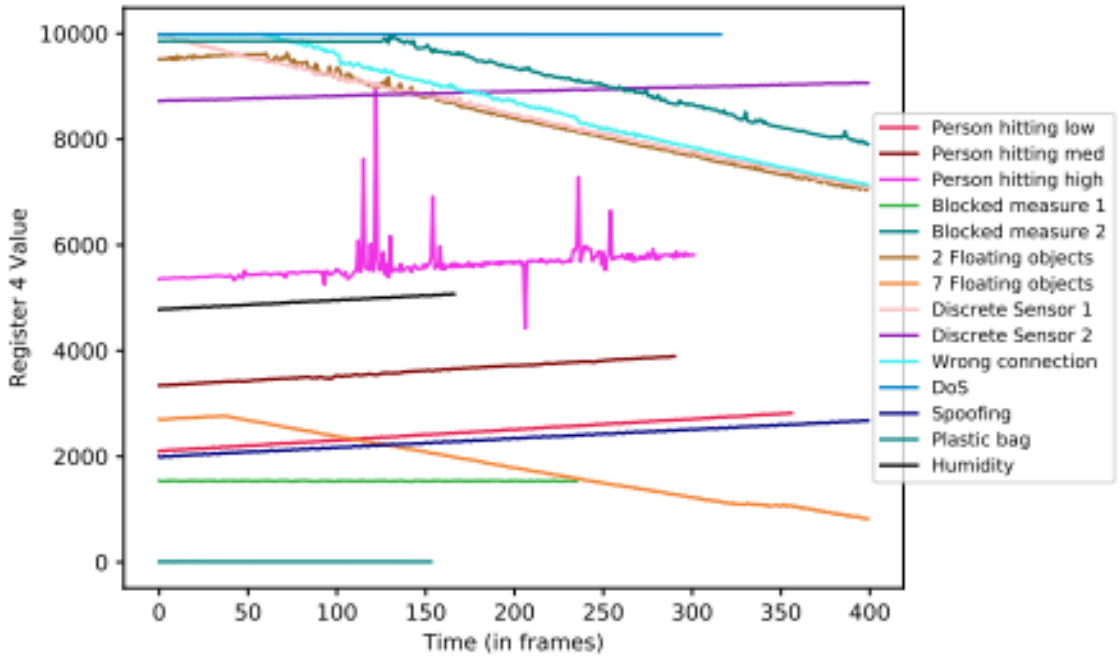


Figure 3.6 SCADA: The trends of the Rate of Change of Register4 readings for different scenarios

The rate of change is determined for each occurrence across a 10-time period, as shown in Equation 3.1.

$$Rate\ of\ change_i = \frac{reg4_i - reg4_{i-10}}{time_i - time_{i-10}}$$

3. Applying Threshold

Table 3.2 shows how the number of instances varies throughout the various circumstances. There is an uneven distribution of occurrences across the scenarios. Classification results will be skewed toward the most common scenario(s) due to training bias. In order to extract only the first N occurrences from each file, a threshold is used. Both (i) keeping the diversity of instances per scenario and (ii) reducing the gap in the number of instances across scenarios are requirements that N should meet.

4. Normalization

If you want your features to be uniform in size and prepared for machine learning, normalization is a must. Using min-max normalization.

5. Cross-Validation

Separating the data into training and test sets is the last step. Using 5-fold cross-validation, the training and testing sets are divided 80/20.

In this project, the datasets are made up of simulated normal and malicious traffic that comes from the traffic creation server. As mentioned before, these datasets are gathered and correlated. We used 20,000 items in the dataset, 10,000 of which were legitimate traffic and 10,000 of which were malicious. The following is a condensed version of an explanation of the three datasets' distinctive features.

Application dataset for the web. The unique aspects of this dataset include information on the TCP and HTTP packets, since it was collected inbound to the webapp, as previously described. In our trials, the tcp characteristics have shown to be quite effective for machine learning.

Datiphy Dataset

In addition to capturing and correlating features such as Response Length and Result returned from the database server to the web application, the Datiphy dataset captures features typical of log analysis, such as the SQL statement that results from the request to the web application. This makes the dataset unique. We don't aware of any other projects that have used such a dataset before this one.

Datasets that are connected. Since this dataset is a merger of the two preceding ones, it incorporates characteristics from both sets; moreover, in our trials, the ML algorithms were able to draw from both sets of features.

In order to test the Siamese network model, four datasets are used. The second one addresses two standard IDS datasets, namely CICIDS2017 and NSL-KDD. In addition, the usefulness of using clean data for training pair generation and adding additional attacks to the trained model is shown by comparing KDD Cup'99 to the NSL-KDD.

To simulate scenarios when data is withheld owing to ethical or privacy considerations, the four datasets are used. In Section 3.3, we get into the first dataset.

3.7 KDD CUP'99

When it comes to assessing intrusion detection systems, the KDD Cup'99 is the gold standard. As previously mentioned, this dataset was used to assess about 60% of the research conducted between 2008 and 2020. Normal activity and four cyber-attack courses make up KDD Cup'99.

The dataset provider releases a reduced subset of around 10% from the rather big KDD Cup'99 dataset. In this experiment, we employ the smaller KDD Cup'99 10% dataset, which includes all classes, to test the suggested Siamese network's suitability for smaller datasets. Instance counts for each class in the KDD Cup'99 dataset are shown in Table 3.3.

Table 3.3 KDD Cup'99 Classes and Corresponding Number of Instances

	Class	# of Instances
1	Normal	97278 (19.70%)
2	DoS	391458 (79.24%)
3	Probe	4107 (0.82%)
4	U2R	1128 (0.23%)
5	R2L	52 (0.01%)

3.7.1 NSL-KDD

In order to address the issues with the KDD Cup'99 dataset that were highlighted by Tavallaee et al., the Canadian Institute for Cybersecurity (CIC) has suggested the NSL-KDD dataset. The four cyber-attack courses and regular activities are both covered in NSL-KDD, same as in KDD Cup'99. Evaluation using the NSL-KDD reveals how improving and filtering a dataset impacts similarity learning and performance. The NSL-KDD dataset's instance counts per class are shown in Table 3.4.

Table 3.4 NSL-KDD Classes and Corresponding Number of Instances

	Class	# of Instances
1	Normal	67343 (53.46%)
2	DoS	45927 (36.47%)
3	Probe	11656 (9.25%)
4	U2R	995 (0.78%)
5	R2L	52 (0.04%)

3.7.2 CICIDS2017

One dataset that the CIC has recently produced is the CICIDS2017 dataset. Recent actual benign, insider, and outer assaults are included in the dataset. The bidirectional traffic flows

are created and labeled using the PCAP files that are given. The number of instances/flows for each assault employed in the trials is listed in Table 3.5. The dataset includes denial-of-service assaults that use various tools to launch them, such as Slowloris and HTTP Unbearable Load King (Hulk).

Table 3.5 CICIDS Classes and Corresponding Number of Instances

	Class	# of Instances
1	Normal	248607 (90.50%)
2	DoS (Hulk)	14427 (5.25%)
3	DoS (Slowloris)	2840 (1.03%)
4	FTP Brute-force	5228 (1.9%)
5	SSH Brute-force	3627 (1.32%)

The NSL-KDD and KDD Cup'99 were preprocessed since they are offered in a feature-like style. After encoding the category features, their 42 features are turned into 118 features in total. In the end, 31 bidirectional flow characteristics were recovered for the CICIDS2017.

It should be emphasized that no feature engineering or selection is done to guarantee that the class that is not trained has no indirect impact on the feature sets.

3.8 SIAMESE NETWORK USAGE SCENARIOS OVERVIEW

Here we take a look at the three main applications of Siamese networks. Figure 3.7 provides a conceptualization of these possibilities.

Assessing similarity learning's viability for cyber-attack categorization is the primary goal of the first scenario. The model is trained using a small subset of cases from all K classes in this case. A new instance P is assigned to one of the K classes after a multi-class classification is carried out.

One-Shot learning is the basis of the second scenario. Instances from K classes are used to train the IDS. Labeled instances from a newly created class e are inserted after training. An extra class e is added to the K training classes, which totals K+1, and a new instance P is

assigned to one of these classes during the evaluation. The effectiveness of the model is assessed by measuring its capacity to categorize both new attack classes and attack classes that were used during training.

Lastly, zero-day attacks—cyberattacks that the model has never seen before and for which there are few labeled cases to fit in the second use scenario—are detected using similarity-based training. In order to determine whether a newly-discovered instance is part of one of the K known classes or an unknown class (i.e., a zero-day assault), the similarity measure is used after training.

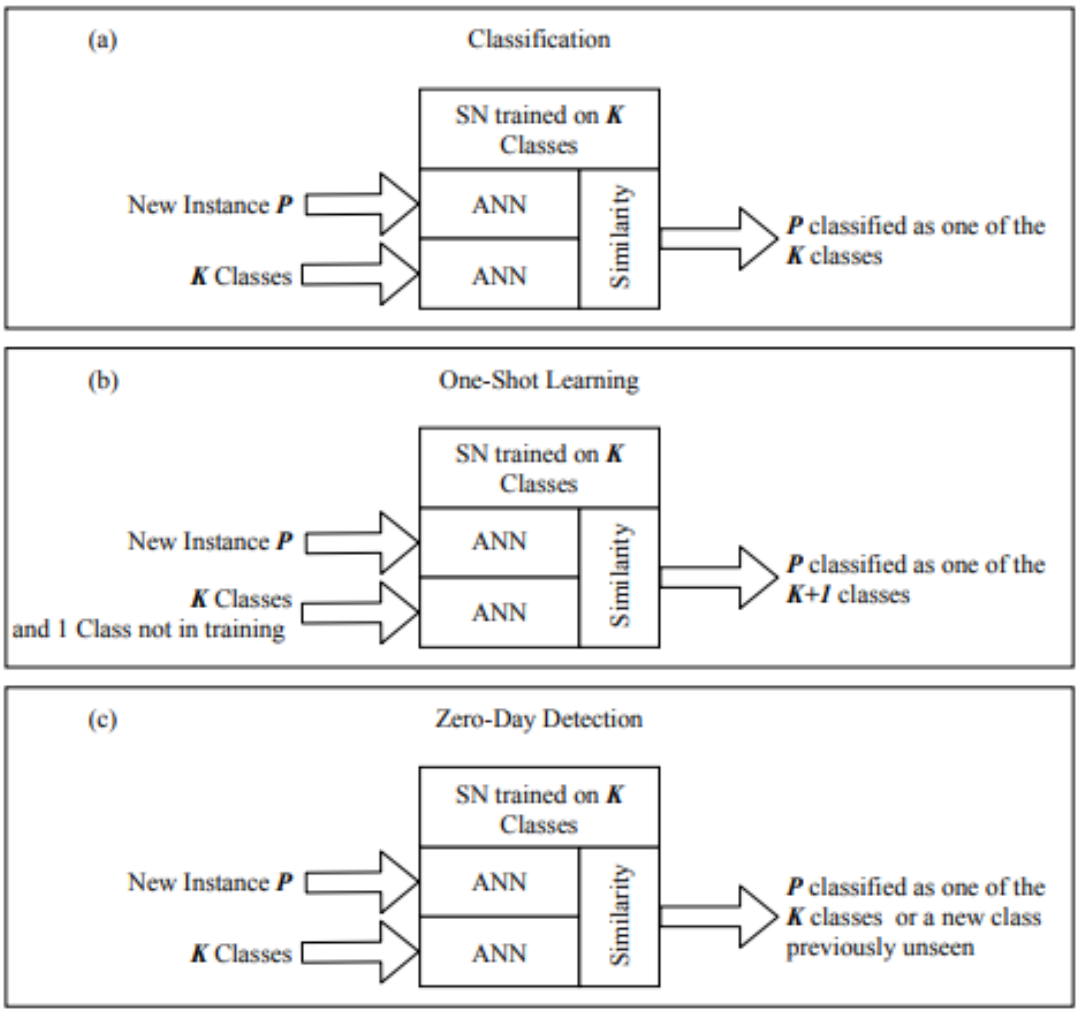


Figure 3.7 Siamese Network Usage Scenarios Overview

3.8 Scenario 1: Classification using Limited Data

Here, we use the Siamese network model to categorize cases from a set of recognized assault types. Pairs that can be retrieved from small samples of each class are used to train

the network. By validating the similarity-based learning paradigm, this decreases the requirement for gathering and annotating vast datasets, which is crucial for IDS development.

The construction of the categorization intrusion detection model is shown in Figure 3.7. As seen in Figure 3.7-1, the dataset is divided into two equal halves. To create comparable and dissimilar pairings for the training and validation sets, the first half is utilized as a training pool of examples (Figure 3.7-2). The instances used to create the testing pairings are taken from the assessment pool in the second part (Figure 3.7-3).

Pairs are selected at random from the pools of instances as the Siamese network model depends on random pair creation. To prevent selection bias in training (when choosing pairings that are similar or different) or assessment (when choosing the labeled examples that best reflect a new class), it is common practice to use instance pools to randomly choose pairs. Plus, there will be no duplicates since each pair is guaranteed to be distinct. Keep in mind that there is no definitive answer to the literature's open issue about the formation of comparable and dissimilar pairings.

In order to conduct testing, N pairings are created by randomly selecting one instance from each class to pair with each instance i. The next step is to determine the label (class) of instance i by comparing all of the chosen pairings and choose the one with the greatest resemblance (i.e., the one with the least distance).

By iteratively matching the examples, we can assess the accuracy of classifications and the quantity of labeled instances required to represent each class.

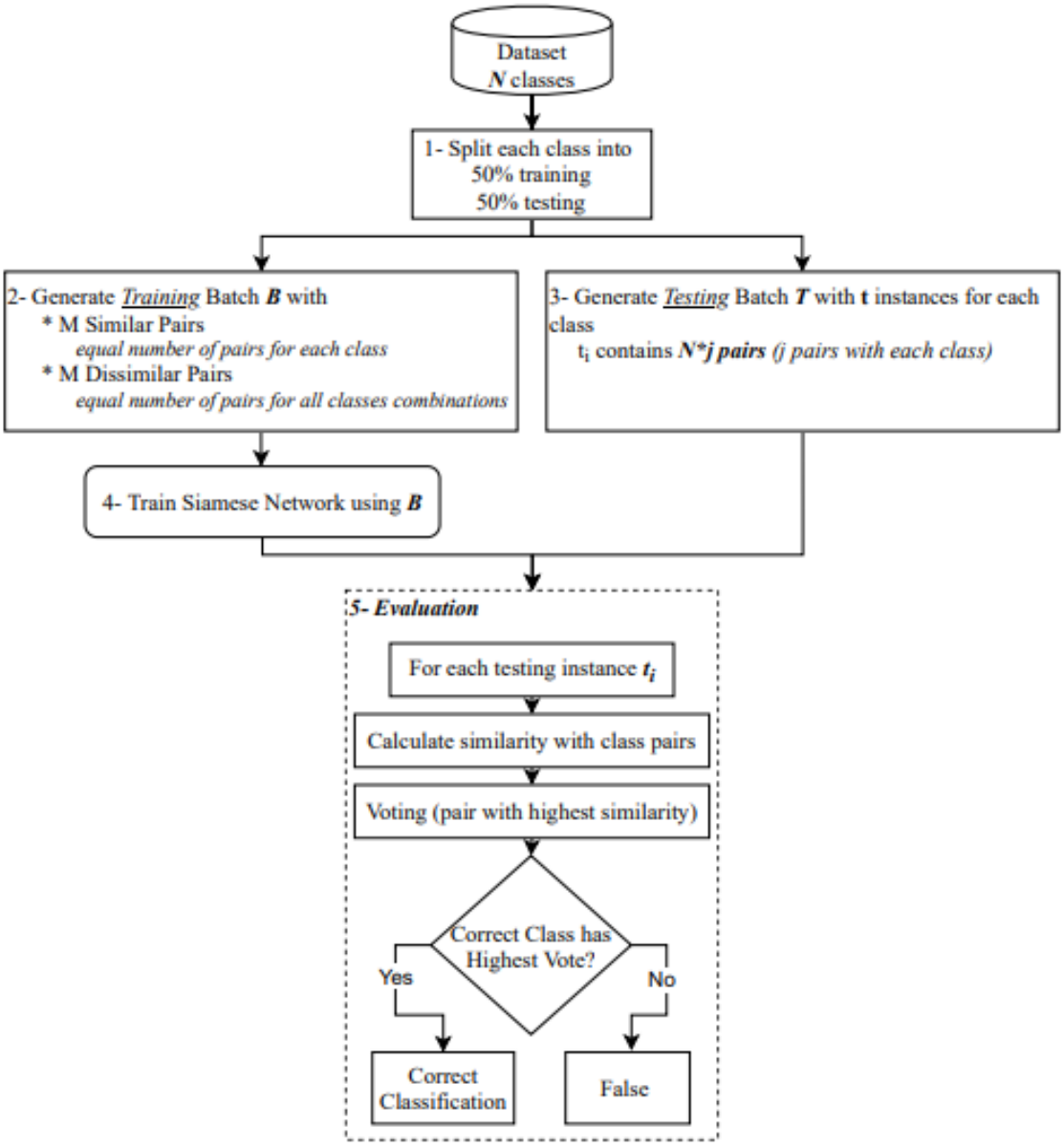


Figure 3.7 Siamese Network for Intrusion Detection (Classification)

iteratively for all occurrences i. The class with the most votes is chosen as the instance i label in a majority voting process (Figure 3.7-5).

Algorithm 3.1 provides a high-level overview of the steps involved in creating pairings, training the Siamese network model, and evaluating it. With a single neuron serving as the output layer and the number of input neurons representing the features, the architecture of a Siamese twin network is defined. Next, we calculate the total number of hidden layers and the number of neurons in each layer. We then set each hidden layer's number of neurons to a fractional value.

Algorithm 3.1 Siamese Network: Usage Scenario 1 Train and Test Algorithm

Input: Attacks Dataset
Output: Trained Siamese Network Evaluation
Ensure: $dataset = \{c_1, c_2, \ldots, c_n\}\ s.th.\ n \geq 3$
1: $train_batch_size, test_batch_size \leftarrow 30,000$
2: $n_epochs \leftarrow 2000$
3: $excluded_class =$ random class e s.th. $e \in dataset$
4: $training_classes = dataset - e$
5: $training = 50\%\ c_i\ \forall c_i \in training_classes$
6: $testing = dataset \cap \overline{training}$
7: $batch \leftarrow$ GETTRAININGBATCH($train_batch_size$)
8: Build Siamese Network with Random Weights
9: **for** $i = 0$ to $n_iterations$ **do**
10: Update Siamese Network Weights based on $batch$
11: **end for**
12: EVALUATE(test_batch_size)

prior to that layer. The design is fine-tuned by optimizing ANN parameters. To make sure the network converges and not overfit during training, we watch the training and validation loss curves using Dropout layers. Based on the model's optimum state, the parameters (such as the number of hidden layers, the number of neurons in each layer, η - learning rate -, number of epochs, etc.) are selected.

The loss behavior may be used to monitor the regularisation of the Siamese network. When training-related instability begins, regularization is applied. The network's performance is erratic, as seen in Figure 3.9.

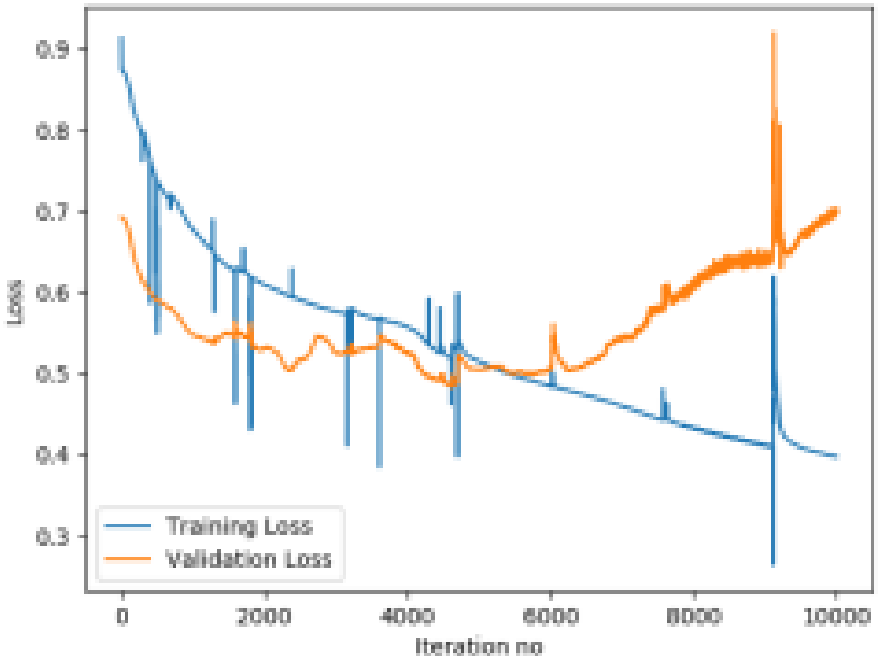


Figure 3.8 Siamese Network Loss Curve (Non-converging case)

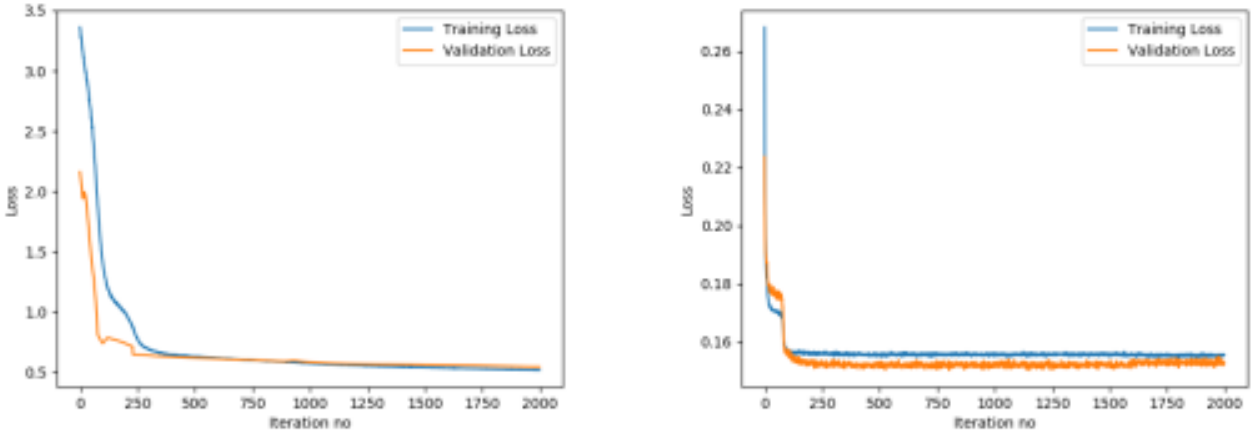


Figure 3.9 Siamese Network Loss Curve (Converging case) - 1

Figure 3.10 Siamese Network Loss Curve (Converging case) – 2

Consequently, in order to prevent overfitting and guarantee network convergence, we reevaluate the network's regularization settings and include dropout layers and kernel regularization. In Figure 3.9 and Figure 3.10, this is clearly seen. Validation of loss curves and optimization of parameters follow the listing of complete model designs in the evaluation section.

Following the steps outlined in Algorithm 3.2, the model is trained using the created batch of pairings for the appropriate number of iterations. Keeping an eye on the loss curves after parameter optimization determines the iteration count, which is 2000 in our example. According to the literature, the recommended batch size for training a Siamese network is 30,000. It should be noted that the classes are evenly distributed throughout the testing and training batches, and that pair uniqueness is assured. Algorithm 3.2 demonstrates how to utilize a "set" data structure to add a pair to the batch of pairs unless that pair is already in the set. If the dataset does not include at least three classes, the Siamese network model will converge to an output of 50% similarity and will not be able to train properly, as stated in the method. This is due to the fact that in the absence of any other classes, the instances have a 50% chance of being either identical (A,A) or different (B,B)). Due to the fact that the unlike pairings are visually similar, the similarity learning process will eventually converge to an output of 50% (0.5 probability).

Algorithm 3.2 Siamese Network: Generate Training Batch

Input: Dataset of N classes, Batch Size
Output: Batch of similar and dissimilar pairs
and associated labels (0: dissimilar, 1: similar)

function GETTRAININGBATCH(batch_size)
    $num_similar_pairs = batch_size/2$
    $num_dissimilar_pairs = batch_size/2$
    $num_similar_pairs_per_class = num_similar_pairs/N$
    $all_combinations = combinations(N)$
    $num_dissimilar_pairs_per_combination$
        $= num_dissimilar_pairs/len(all_combinations)$
    $pairs_set \leftarrow \{\}$
    for c in N **do**
        for $i = 0$ to $num_similar_pairs_per_class$ **do**
            $(ins_1, ins_2) \leftarrow$ 2 random instances $\in c_training$
            if $(ins_1, ins_2) \in pairs_set$ **then**
                go to 10
            end if
            $pairs[i] \leftarrow \{ins_1, ins_2\}$
            $pairs_set.add(\{ins_1, ins_2\})$
        end for
    end for
    for c_1, c_2 in $all_combinations$ **do**
        for $i = 0$ to $num_dissimilar_pairs_per_combination$ **do**
            $ins_1 \leftarrow$ random instance $\in c_1_training$
            $ins_2 \leftarrow$ random instance $\in c_2_training$
            if $(ins_1, ins_2) \in pairs_set$ **then**
                go to 20
            end if
            $pairs[i] \leftarrow \{ins_1, ins_2\}$
            $pairs_set.add(\{ins_1, ins_2\})$
        end for
    end for
    $targets[0..batch_size/2] \leftarrow 1$ ▷ Similar
    $targets[batch_size/2..batch_size] \leftarrow 0$ ▷ Dissimilar
    return $pairs, targets$
end function

An equal number of instances from each class are utilized for the evaluation (Algorithm 3.3). Each class has a pair that is chosen for each new instance by taking the new instance and picking an instance at random from that class. Each set of two items is compared for their degree of similarity; the set that has the highest degree of similarity is used to determine the category. This procedure is carried out j times and the outcomes are compiled via voting, where j ranges from 1 to 30.

Algorithm 3.3 Siamese Network: Evaluate Classification

Input: Trained Siamese Network, Batch Size
Output: Classification Accuracy

function EVALCLASSIFICATION(batch_size)
    $n_correct \leftarrow 0$
    $num_per_class \leftarrow batch_size/N$
    for c in N **do**
        for $i = 0$ to num_per_class **do**
            $ins_1 \leftarrow$ random instance $\in c_testing$
            for $j = 0$ to 5 **do**
                $pairs \leftarrow (ins_1,$ random instance $x \forall x \in N)$
                $similarities \leftarrow model.predict(pairs)$
                $votes[argmin(similarities)] += 1$
            end for
            if $argmax(votes) == c$ **then**
                $n_correct += n_correct + 1$
                $confusion_matrix[c, argmax(votes)] += 1$
            end if
        end for
    end for
    $accuracy = n_correct * 100/batch_size$
    return $accuracy, confusion_matrix$
end function

3.8 OUTLIER-BASED ZERO-DAY ATTACKS DETECTION

In order to test the models discussed in this chapter, three popular IDS datasets were selected. The datasets included for examination are CICIDS2017, NSL-KDD, and KDD Cup'99.

Over the course of five days, the CICIDS2017 dataset [63] records a diverse array of interior and external assaults that have occurred recently. It is offered in a raw format that permits freedom in processing the information, and it offers a broad coverage of protocols and attack variants. The daily traffic situations are summarized in Table 3.6. As will be explained later on, the CICIDS2017 dataset's raw files undergo preprocessing. Refer to [325] for the whole CICIDS2017 description.

Second, there is the KDD Cup'99 [78] dataset, and finally, there is the NSL-KDD [77]. The CSV feature files include two datasets that include four types of cyber attacks: DoS, Probing, R2L, and U2R, in addition to typical traffic. Along with the class name, each

instance is represented by its feature values. In order to get the feature files ready for machine learning, they are categorical feature encoded.

Table 3.6 CICIDS2017 Attacks

Day	Traffic
Monday	Benign
Tuesday	SSH and FTP Brute-force
Wednesday	DoS/DDoS and Heartbleed
Thursday	Web Attack (Brute-force, XSS, Sql Injection) and Infiltration
Friday	Botnet, Portscan and DDoS

In order to train the models, only benign traffic events are used. The sklearn "train test split" function [326] is used to divide the benign cases into training and validation sets. Next, we test the model's detection capabilities by simulating a zero-day assault using each of the attack types. Because the NSL-KDD dataset comes in two files, "KDDTrain+.csv" and "KDDTest+.csv," we test against attacks in both of them.

3.8.1 CICIDS2017 Dataset Preprocessing

The following procedure is described to get the CICIDS2017 dataset ready. To begin, the CICIDS2017 dataset's PCAP files are categorized according to the kind of assault and the timestamps supplied by the dataset's owner. Consequently, every kind of assault gets its own PCAP file. Second, characteristics of bidirectional flows are retrieved. Developers of contemporary intrusion detection systems would do well to stick to flow-based features. The reliance on encrypted communication, together with the increasing sophistication and size of networks, is to blame. Because they are derived from the connection between two nodes rather than particular packet contents, flow-based characteristics may be used for both encrypted and unencrypted traffic analysis. The third step in reducing model instability is to remove characteristics that have a strong association.

Algorithm 3.4 explains how to remove characteristics that are very linked. A cutoff value of 0.9 is used. Training features are those whose correlation is lower than the threshold. This is due to the fact that qualities that are highly correlated are likely to influence the dependent variable in a comparable way [328]. It follows that one of them must be abandoned. Lastly, a Standard Scaler is used to scale the features. To speed up the training

process overall, this is done to normalize the features to a mean of μ of 0 and a standard deviation of σ. For the purpose of feature selection and scalability, we use only benign instances to guarantee that cyber attack instances have no impact.

Algorithm 3.1 Drop Correlated Features

Input: Benign Data 2D Array, N, Correlation Threshold
Output: Benign Data 2D Array, Dropped Columns

1: $correlation_matrix \leftarrow data.corr().abs()$
2: $upper_matrix \leftarrow correlation_matrix[i,j] \qquad \{i,j \in N : i <= j\}$
3: $dropped \leftarrow i\{i \in N : correlation_matrix[i,^*] > threshold\}$
4: $data \leftarrow data.drop_columns(dropped)$
5: **return** $data, dropped$

3.8.2 Autoencoder-based model

The suggested autoencoder is constructed using an ANN. Optimizing the hyperparameters of an ANN makes use of random search [275]. It is from the results of the random search that the ANN design, learning rate, and amount of epochs are determined. In order to find a semi-optimal collection of parameters, random search is believed to converge more quickly than grid search. It outperforms grid search when just a few parameters are required, as previously shown [329]. Last but not least, it reduces the likelihood of overfitted parameters.

The training of the model begins when the hyper parameters have been determined. An overview of the training procedure is provided in Algorithm 3.2. At the outset, we divide the benign examples as follows: 75% for training and 25% for validation [231]. At startup, the model is trained using the best ANN design (layer count and hidden neuron number).

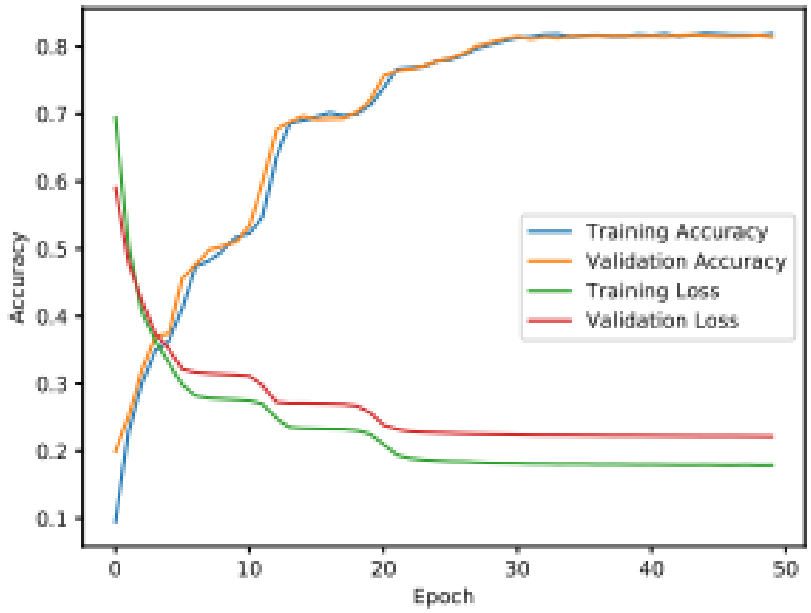


Figure 3.11 Autoencoder Convergence Curve

per layer). Lastly, the model undergoes training for an n-epoch sequence. It is verified that the auto encoder converges by analyzing the loss and accuracy curves.

Algorithm 3.3 is used to assess the model once it converges, as seen in Figure 3.11. If there is a significant difference between the decoded and original instances, as measured by the Mean Squared Error (MSE), then the assault instance is considered a zero-day attack. First, the value given by the random search hyper parameter optimization is used to choose the threshold. Various criteria, including 0.05, 0.1, 0.15, and 0.2, are evaluated for this purpose. An instance is deemed a zero-day attack at a certain value, defined here as the MSE that falls inside the permitted range between X 0 and X, and this value is heavily influenced by the threshold.

Algorithm 3.2 Autoencoder: Training

Input: benign_data, ANN_architecture, regularisation_value, num_epochs
Output: Trained Autoencoder

1: $training = 75\%$ of $benign_data$
2: $testing = benign_data - training$
3: $autoencoder \leftarrow build_autoencoder$(ANN_Architecture, regularisation_value)
4: $batch_size \leftarrow 1024$
5: $autoencoder.train(batch_size, num_epochs, training, testing)$
6: **return** $autoencoder$

Algorithm 3.3 Autoencoder: Evaluation

Input: Trained Autoencoder, attack, thresholds
Output: Detection accuracies

1: $detection_accuracies \leftarrow \{\}$
2: $predictions \leftarrow model.predict(attack)$
3: **for** $th \in thresholds$ **do**
4: $\quad accuracy \leftarrow (mse(predictions, attack) > th)/len(attack)$
5: $\quad detection_accuracies.add(threshold, accuracy)$
6: **end for**
7: **return** $detection_accuracies$

3.8.3 One-Class SVM based Model

One-Class SVM models, like autoencoder-based models, are trained using only benign cases. The One-Class SVM is trained using a "ν" value. In their description of the ν value, Chen et al. state that it is "ν ∈ [0, 1] which is the lower and upper bound on the number of examples that are support vectors and that lie on the wrong side of the hyperplane, respectively." [330]. The scikit-learn library has 0.5 as its default value for ν [227]. In other words, we want to make sure that half of the training samples fall on a certain hyperplane. Nevertheless, a range of ν values (0.2, 0.15, 0.1, 0.05) was used for this experiment. The autoencoder's performance was assessed and compared using these ν values.

To train the One-Class SVM model, refer to Algorithm 3.4. The training method uses 75% of the benign examples, same as the autoencoder model in Section 3.2.1. In contrast to the autoencoder model, which uses a threshold for assessment, a trained One-Class SVM model produces a one-of-two binary result. The output shows whether an instance is a member of the class that the One-Class SVM is trained on. The number of cases predicted with a "0" output of the One-Class SVM is therefore used to assess each assault.

Algorithm 3.4 One-Class SVM Model

Input: benign_data, nu_value
Output: Trained SVM

1: $training = 75\%\ i \in benign_data$
2: $testing = benign_data \cap \overline{training}$
3: $oneclasssvm \leftarrow OneClassSVM(\ nu_value, `rbf')$
4: $oneclasssvm.fit(training)$
5: **return** $oneclasssvm$

CHAPTER 4

EXPERIMENT AND RESULTS

4.1 OVERVIEW

Table 4.1 summarizes the results of our research with several machine learning techniques. The time it takes to construct the machine learning models is called Model Time, the time it takes to classify the testing dataset using 5-fold cross-validation is called Testing Time, and Accuracy is the classification accuracy that each method achieves. Figure 4.1 displays the classification accuracy, whereas Table 4.2 displays the F-scores.

Table 4.1: Results With 20000 Records

Dataset	Algorithm	Accuracy	Model Time	Testing Time
Webapp	JRip	94.740%	3m30.85s	2.70s
	J48	95.630%	1m42.65s	2.55s
	RF	96.525%	5m30.20s	30.60s
	SVM	94.025%	2m35.80s	1m10.45s
	ANN	96.715%	46m03.55s	3.35s
Datiphy	JRip	95.980%	6m10.90s	3.35s
	J48	96.995%	1m59.30s	2.45s
	RF	97.210%	6m10.20s	33.50s
	SVM	95.190%	2m11.50s	1m3.45s
	ANN	97.285%	41m23.00s	2.95s
Correlated	JRip	97.150%	11m50.80s	2.10s
	J48	97.295%	2m01.80s	2.05s
	RF	98.055%	4m22.55s	35.45s
	SVM	95.715%	3m30.85s	1m5.90s
	ANN	97.615%	47m25.25s	3.80s

Table 4.2: F-score for 20000 Record Data

Dataset	Algorithm	True Pos.	False Pos.	True Neg.	False Neg.
Webapp	JRip	9117	169	9831	883
	J48	9385	259	9741	615
	RF	9487	182	9818	513
	SVM	9134	329	9671	866
	ANN	9501	158	9842	499
Datiphy	JRip	9291	95	9905	709
	J48	9538	139	9861	462
	RF	9757	46	9954	243
	SVM	9226	188	9812	774
	ANN	9613	156	9844	387
Correlated	JRip	9623	193	9807	377
	J48	9656	197	9803	344
	RF	9860	32	9986	140
	SVM	9457	314	9686	543
	ANN	9644	121	9879	356

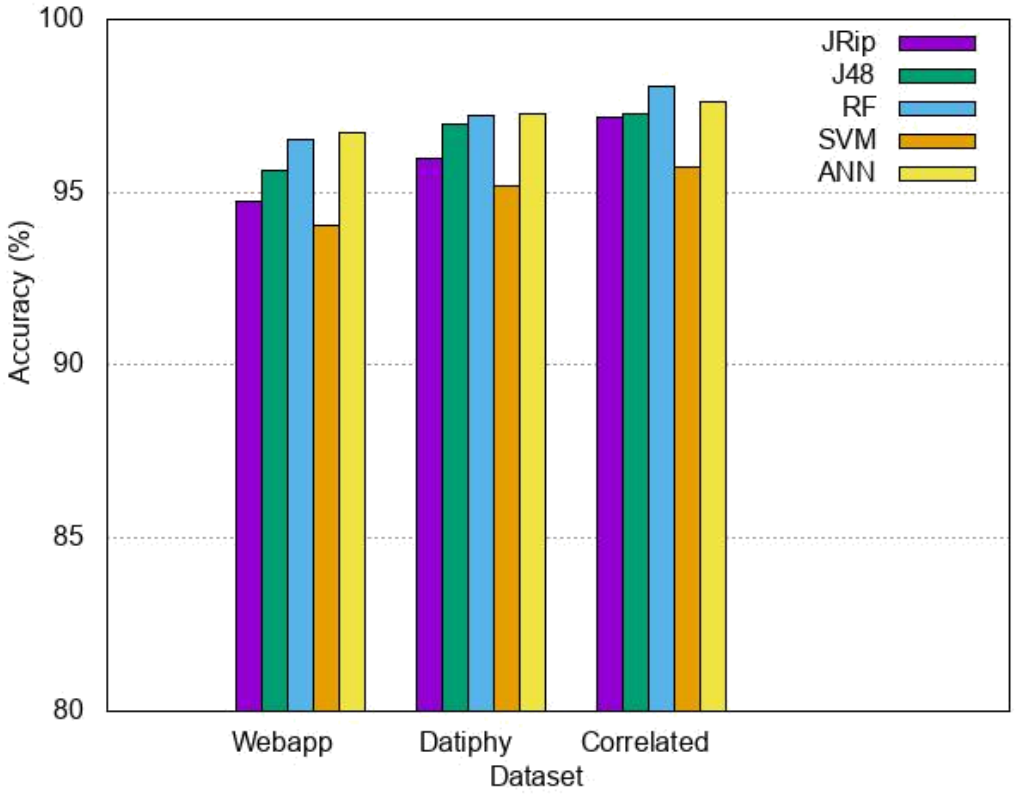


Figure 4.1: Classification Accuracy

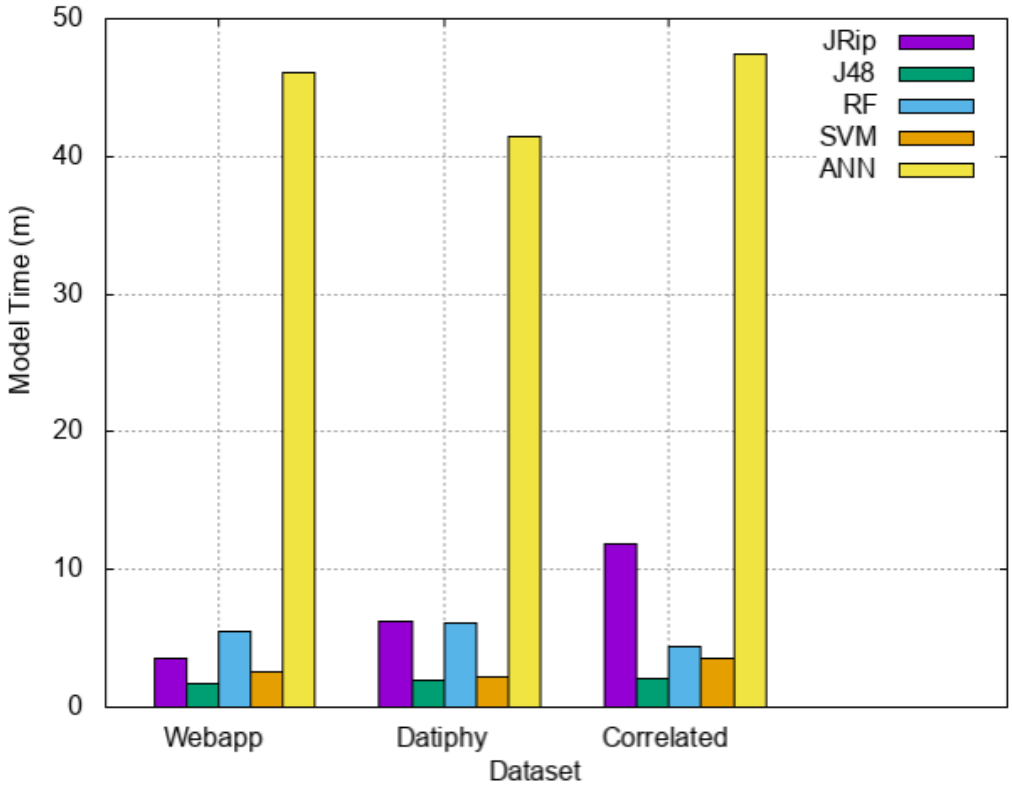


Figure 4.2: Modeling Time

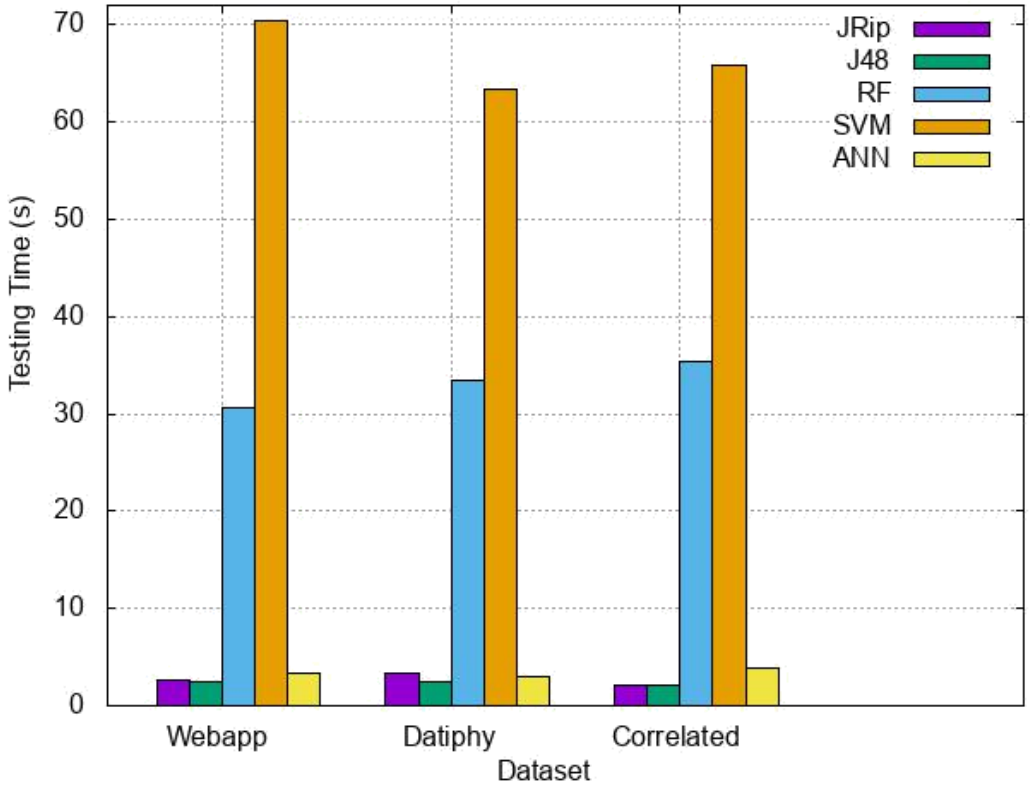


Figure 4.3: Testing Time

Since the correlated dataset is a mixture of two datasets, it contains more data than the individual datasets, which, according to my intuitive understanding of the procedure, should provide superior classification accuracy results. It has been the case so far in our research; as previously said, JRip is one example that uses characteristics from both databases to generate rules. Take the JRip algorithm's top-ranking criteria as an example; they combine two datasets—the webapp dataset for http. content length and the Datiphy dataset for SQL Length.

We benchmarked the JRip and J48 algorithms using datasets consisting of 2000 records in our original findings. All of these findings are consistent with the idea that the linked datasets provide much better outcomes than each of the separate datasets in isolation. As a consequence, we can say that bigger datasets consistently provide better outcomes across all measures. We do use randomness in our attack creation, however it seems like the data is fitting into categories that the machine learning algorithms can use to build a model. Their ability to do so becomes better with more data. In this research, we have integrated the SVM, Random Forest, and ANN algorithms and documented their performance.

We found that the decision tree method performed quite well in our tests, which is an important part of the performance findings. It is commonly believed that decision tree algorithms are resource intensive. However, we hypothesized that decision tree algorithms outperform other classification algorithms when the data is typically easy to classify using a small number of carefully chosen features. This hypothesis was tested in the present project.

We found that the quicker methods provide results on the correlated dataset that are almost identical to our MultiLayer Perceptron findings, but take much less time to execute, based on our trials. According to one conclusion drawn from these studies, using data from several sources may enhance classification results just as much, if not more so, than using algorithms that may be more accurate.

4.2 SCADA

To demonstrate the precision of anomaly detection, this section details and assesses three separate studies. The purpose of the various tests is to offer varying degrees of detail about the presence of an abnormality. From just noting that an abnormality has occurred to pinpointing the offending part and the unusual circumstance, there is a wide range of possible outcomes.

4.2.1 Experiment 1: Anomaly Detection

In actual use, intrusion detection systems should trigger an alarm in the event of an abnormality. In the first test, we see how well the six ML algorithms we covered before can detect and report abnormalities in the SCADA dataset. Since the models are used as binary classifiers to differentiate between abnormal and normal behavior, this experiment does not identify the kind of anomaly or the component that is impacted, unlike the ones that follow.

The various ML algorithms' classification accuracies are shown in Figure 4.3. The results show that the best accuracies achieved with k-NN, RF, DT, and SVM with RBF kernel are 94.12%, 93.67%, 93.30%, and 91.99%, respectively.

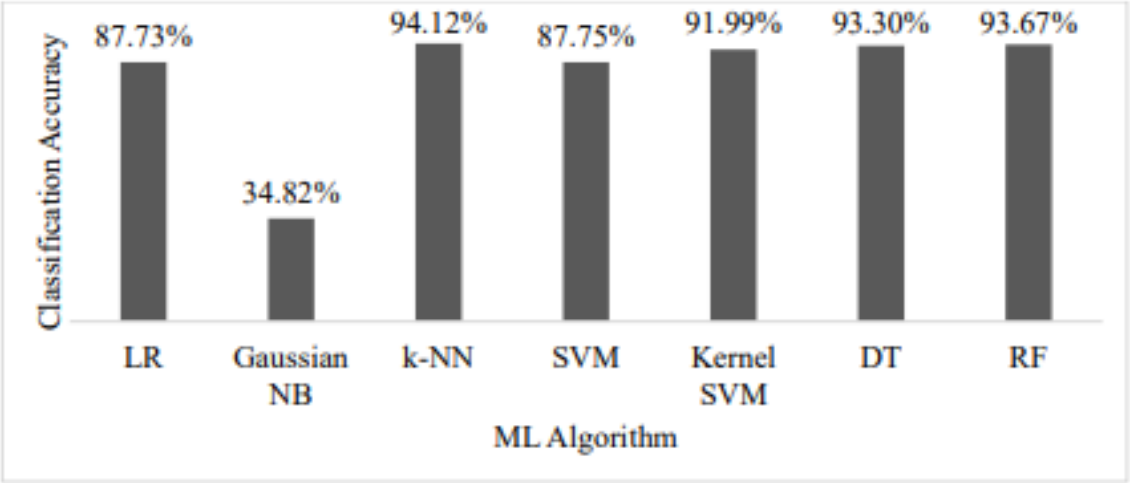


Figure 4.2 SCADA: Anomaly Detection Overall Accuracy (5-fold cross-validation)

You can find all the information about the recall, precision, and F1-Score from the first experiment in Table 4.3. In terms of overall accuracy, the top three methods (k-NN, DT, and RF) can effectively distinguish between normal and abnormal behavior. The recall for the benign class is 74.01% with k-NN, 96.22% with DT, and 96.38% with RF, whereas for the anomalous class it is 97.15%, 96.22%, and 96.38%, respectively. These three algorithms are superior than others because, unlike those that try to match a hyperplane or a probabilistic connection, they are able to map complicated relations and non-linearity. Class separability becomes more complicated in a CI environment since regular operations and abnormalities might coexist.

Table 4.3 SCADA Results: Experiment 1 - Anomaly Detection (5-fold cross-validation)

Classification (Is Anomaly)	Recall	Precision	F1-Score
LR			
Benign	7.15%	90.34%	13.22%
Anomaly	99.89%	87.7%	93.4%
Weighted Average	87.73%	88.05%	88.05%
NB			
Benign	99.95%	16.74%	28.67%
Anomaly	24.99%	99.97%	39.98%
Weighted Average	34.82%	89.06%	89.06%
k-NN			
Benign	**74.01%**	79.7%	76.74%
Anomaly	**97.15%**	96.12%	96.63%
Weighted Average	**94.12%**	93.97%	93.97%
SVM			
Benign	7.15%	92.24%	13.23%
Anomaly	99.91%	87.7%	93.41%
Weighted Average	87.75%	88.3%	88.3%
Kernel SVM			
Benign	39.53%	98.52%	56.4%
Anomaly	99.91%	91.63%	95.59%
Weighted Average	91.99%	92.54%	92.54%
DT			
Benign	**74.01%**	74.72%	74.35%
Anomaly	**96.22%**	96.09%	96.15%
Weighted Average	**93.3%**	93.28%	93.28%
RF			
Benign	**75.66%**	75.99%	75.78%
Anomaly	**96.38%**	96.33%	96.36%
Weighted Average	**93.67%**	93.67%	93.67%

Although it is crucial to trigger an alert in the event of an anomaly, the warning is presented in a binary format, making it difficult to immediately diagnose the issue. Hence, a move to rectify the situation is postponed. In pursuit of this goal, a second experiment is set up to gather more data.

4.2.2 Experiment 2: Affected Component Classification

The goal of the second experiment is to use multi-class classifiers to report the impacted component whenever an anomaly occurs, rather than just shooting an alarm. When compared to the first experiment, this new piece of data gives more specifics, allowing for faster course corrections.

The classification results of the various ML algorithms are shown in Figure 4.4. The best accuracy rates achieved using k-NN, RF, and DT are 82.69%, 82.71%, and 81.79 percent, respectively. There was a compromise between general correctness and specificity, as shown by the outcome.

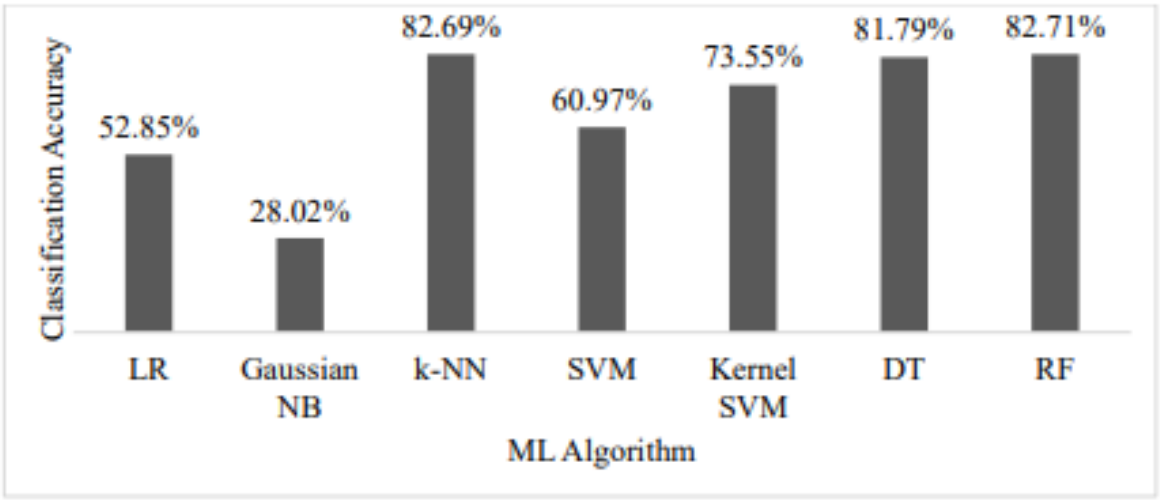


Figure 4.3 SCADA: Affected Component Overall Classification Accuracy (5-fold cross-validation)

in comparison to the binary classification described in Section 4.4.1 and the alarm that was triggered. A high proportion of false positives are accounted for by the model's accuracy since more classes are involved. The recall, accuracy, and F1-Score for each method are detailed in Table 4.4, where this trade-off is clearly highlighted. At 39.83% and 59.82% respectively, LR and SVM had the worse recall. This is because the classes of impacted components are non-linear. Thus, these methods do not fit, even if k-NN, DT, and NB all achieve recalls in the benign class of 79.76%, 74.31%, and 79.76%, respectively.

Table 4.4 SCADA Results: Experiment 2 - Affected Component Classification (5-fold cross-validation)

Classification (Affected Component)	Recall	Precision	F1-Score
LR			
None	39.83%	31.38%	35.08%
Discrete Sensor 1	59.53%	37.16%	45.75%
Discrete Sensor 2	23.79%	60.17%	34.05%
Network	56.57%	53.66%	55.06%
Ultrasound Sensor	62.55%	70.09%	66.09%
Whole	69.57%	100%	81.98%
Weighted Average	52.85%	56.74%	56.74%
NB			
None	79.76%	38.48%	51.9%
Discrete Sensor 1	50.57%	35.71%	41.82%
Discrete Sensor 2	0%	0%	0%
Network	16.71%	21%	18.61%
Ultrasound Sensor	1.73%	98.89%	3.39%
Whole	100%	18.82%	31.67%
Weighted Average	28.02%	43.26%	43.26%
k-NN			
None	74.46%	78.39%	76.33%
Discrete Sensor 1	76.82%	74.17%	75.43%
Discrete Sensor 2	68.47%	63.09%	65.66%
Network	84.73%	87.45%	86.05%
Ultrasound Sensor	90.55%	90.59%	90.57%
Whole	99.13%	100%	99.56%
Weighted Average	82.69%	83%	83%
SVM			
None	59.82%	42.95%	50%
Discrete Sensor 1	84.69%	58.63%	69.25%
Discrete Sensor 2	23.79%	57.98%	33.69%
Network	54.27%	55.87%	55.05%
Ultrasound Sensor	66.83%	72.89%	69.72%
Whole	100%	100%	100%
Weighted Average	60.97%	62.05%	62.05%

Kernel SVM			
None	62.82%	67.27%	64.96%
Discrete Sensor 1	82.24%	61.62%	70.43%
Discrete Sensor 2	61.62%	51.93%	56.36%
Network	70.54%	79.86%	74.9%
Ultrasound Sensor	77.84%	85.76%	81.59%
Whole	99.57%	100%	99.78%
Weighted Average	73.55%	75.06%	75.06%
DT			
None	74.31%	75%	74.63%
Discrete Sensor 1	73.7%	74.46%	74.06%
Discrete Sensor 2	64.72%	64.11%	64.39%
Network	84.8%	85.29%	85.04%
Ultrasound Sensor	90.3%	89.4%	89.85%
Whole	99.89%	100%	99.95%
Weighted Average	81.79%	81.8%	81.8%
RF			
None	74.16%	78.24%	76.11%
Discrete Sensor 1	77.19%	73.68%	75.38%
Discrete Sensor 2	68.32%	64.32%	66.22%
Network	85.12%	86.43%	85.77%
Ultrasound Sensor	90.06%	91.18%	90.61%
Whole	99.89%	100%	99.95%
Weighted Average	82.71%	82.95%	82.95%

Although there is a wide variety of events that might impact many components, this experiment's findings show that some are more difficult to identify than others due to class overlap. Using LR, NB, k-NN, SVM, Gaussian SVM, DT, and RF, in that order, yields a low recall of 23.79%, 0%, 68.47%, 61.62%, 64.72%, and 68.32% for the "Discrete Sensor 2" class detection, respectively. Conversely, the "Whole" class has excellent recall. In this example, the recall for LR, NB, k-NN, SVM, Gaussian SVM, DT, and RF is 69.57%, 100%, 99.13%, 100%, 99.57%, and 99.89%, respectively.

The model's reporting capabilities are enhanced by identifying the impacted component. It facilitates a more precise diagnosis of the issue, enabling a more expedited resolution. On the other hand, detailing the precise situation would be ideal. Because of this, we can find

the issue faster and put a stop to it. Consequently, a third experiment is set up to achieve this goal.

4.2.3 Experiment 3: Scenarios Classification

Extending the system's capacity to detect suspicious situations and flag them is the objective of the third experiment. Three operational trials are carried out to carry out this experiment. In order to identify the potentially dangerous situation, the studies use multi-class classifiers and utilize the probabilities of their outputs.

4.2.3.1 One Scenario Classification

The first run involves training ML models to categorize various situations according to the multi-class classifier's greatest likelihood. The outcomes of the various ML methods are shown in Figure 4.5. Even at its peak, accuracy is just 81.19%. Appendix C contains the entire results tables, including recall, precision, and F1-Score, for the sake of completeness and repeatability.

For many classes, the findings show a high or poor recall. The following findings are reached after reviewing the situations in order to analyze this:

(i) The dataset contains situations that are co-related, which does not rule out the possibility of overlap. We are aware of this issue with the CI setup.

(ii) For any given case, the models provide at least two and no more than four possible outcomes.

Further explanation of the second finding is that there can be no more than four non-zero scenario probabilities for each occurrence since ML models can output the classification outcome probability. Thus, there is a limit of four possible scenarios to which any instance might belong. For example, Table 4.6 shows the total number of cases with several alternative outcomes along with their percentages. Since the

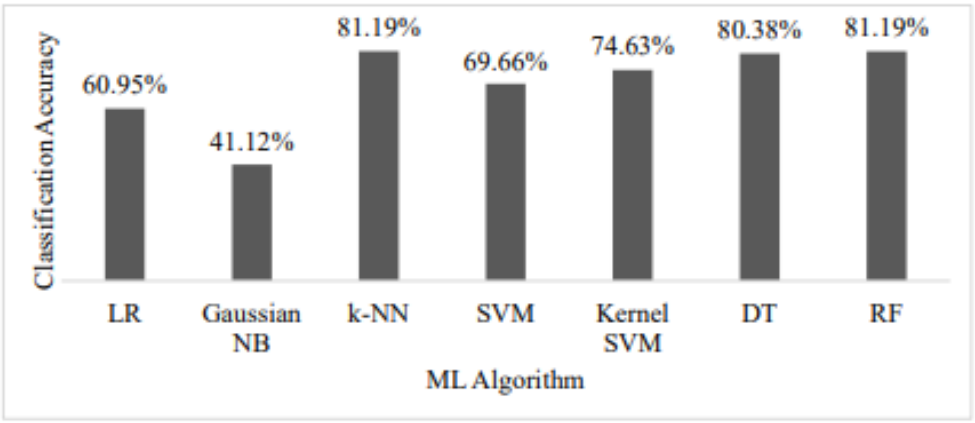


Figure 4.5 SCADA: Scenarios Overall Classification Accuracy, Single Scenario (5-fold cross-validation)

Table 4.5 SCADA: Distribution of Probabilistic Classification of Scenarios

Algorithm	Maximum number of probable scenarios per instance	Number of instance with **1** probable scenario	Number of instance with **2** probable scenarios	Number of instance with **3** probable scenarios	Number of instance with **4** probable scenarios
DT	1	3817	-	-	-
		100%	-	-	-
k-NN	3	2336	1221	260	-
		61.20%	31.99%	6.81%	-
RF	4	2144	1315	355	3
		56.17%	34.45%	9.3%	0.08%

maximum of four possible outcomes; the table displays the number of occurrences with one, two, three, or four outcomes. According to the data in the second row, for instance, 61.2% of the occurrences are categorized with a single likely class, 31.99% with two, and 6.81% with three. Consequently, in order to lessen the ambiguity around this method, the following experiments take use of this to provide two likely outcomes.

4.2.3.2 Two Scenarios Classification

When an anomaly happens in the second trial, the model returns two possibilities instead of one in the first experiment. The two cases represent the most likely outcomes according to the various classifiers. Since the two reported situations overlap and CI anomalies are

not mutually exclusive, an occurrence is appropriately identified if it corresponds to one of the two. Here, the detection accuracy might be enhanced by using multi-label classification. Figure 4.7 indicates that total detection accuracy rises to 95.60% and 95.49% when employing RF and k-NN, respectively, due to this adjustment. Better action is possible in light of the stated cases with a more accurate detection model. This lessens the severity of the assault and shortens the time required for a whole reaction.

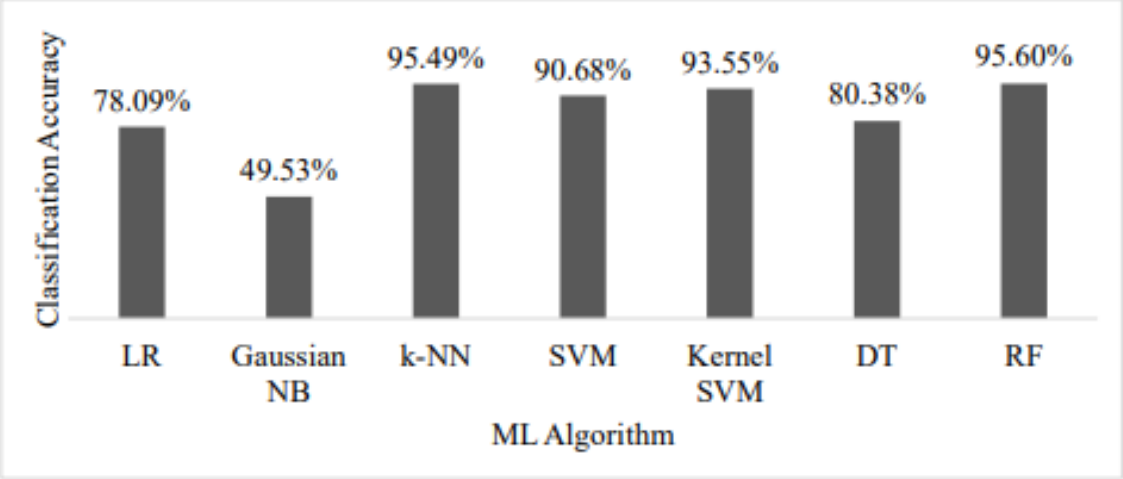


Figure 4.6 SCADA: Scenarios Overall Classification Accuracy, Two Probable Scenarios (5-fold cross-validation)

Instances that are appropriately labeled when one likely scenario is presented vs two likely scenarios are shown in Table 4.6. The above example is based on calculations made using the k-NN classification model. For whatever reason, 67 of the objects in the first row in the "2 Floating Objects" scenario should be considered "Plastic Bag" sabotage. Still, taking the second most likely into account allows for the accurate reporting of sixty of them.

Table 4.6 SCADA: Co-relation of scenarios that are misclassified based on one probable scenario and correctly reported with the second probable one (Calculated based on k-NN experiment)

Scenario (X)	Instances count where X		Scenario (Y) The count of instances classified as Y while the correct is X						
	Is Not 1st Probable Scenario	**Is** 2nd Probable Scenario	2 Floating Objects	7 Floating Objects	Normal	Plastic Bag	Sensor Failure	Spoofing	Wrong Con.
2 Floating Objects	67	60	-	-	-	67	-	-	-
7 Floating Objects	5	5	-	-	-	-	-	-	5
Normal	113	78	-	-	-	1	96	-	16
Plastic Bag	107	91	49	-	3	-	35	20	-
Sensor Failure	242	184	-	-	99	73	-	9	61
Spoofing	54	32	-	-	-	44	10	-	-
Wrong Con.	134	96	-	7	31	8	88	-	-

scenario. Similarly, in row 4 of Table 4.6, there are 107 cases of the "Plastic Bag" scenario that are incorrectly labeled as "2 Floating Objects" (49 cases), "Sensor Failure" (35 cases), "Spoofing" (20 cases), and "Normal" (3 cases). Taking the second most likely situation into account allows for the accurate reporting of 91 of these events.

We present two possible outcomes from this experiment. When the first case is adequate, this could lead to erroneous conclusions. This is the result when there is more than one possible explanation for an anomaly. Including a second scenario in the report would be superfluous here. As a result, the situational handling reaction is enhanced thanks to the third trial's confidence measure, which enables improved reporting.

4.2.3.3 Scenarios Classification Using Confidence

Unless the scenario's categorization probability falls below a certain level, just one scenario is reported in the third trial. As a model confidence interval, this cutoff value is useful. Accordingly, two cases are detailed if the model's classification probability drops below that cutoff. The experiment was conducted with two threshold values—75% and 85%—and the results are shown in Figure 4.7.

Use of a 75% confidence interval results in the reporting of one scenario when the classification probability is at least 0.75 and the reporting of two possibilities otherwise. A total of 91.57% of detections are accurate. Using a confidence interval of 85% improves the overall accuracy of the categorization. In Figure 4.8, this is shown. Using k-NN, the total detection accuracy may reach a maximum of 95.49%.

Figure 4.6 and Figure 4.7 allow for the deduction of certain conclusions. The reason why DT accuracy is consistent across tests is because DT provides a single scenario for each occurrence based on the formation of the decision branches. In every single case, the DT model produces a single scenario, as can be shown in Table 4.5. Consequently, reporting two potential outcomes is not feasible. With an accuracy of 95.39 percent, the assault may be quickly mitigated with less ambiguity.

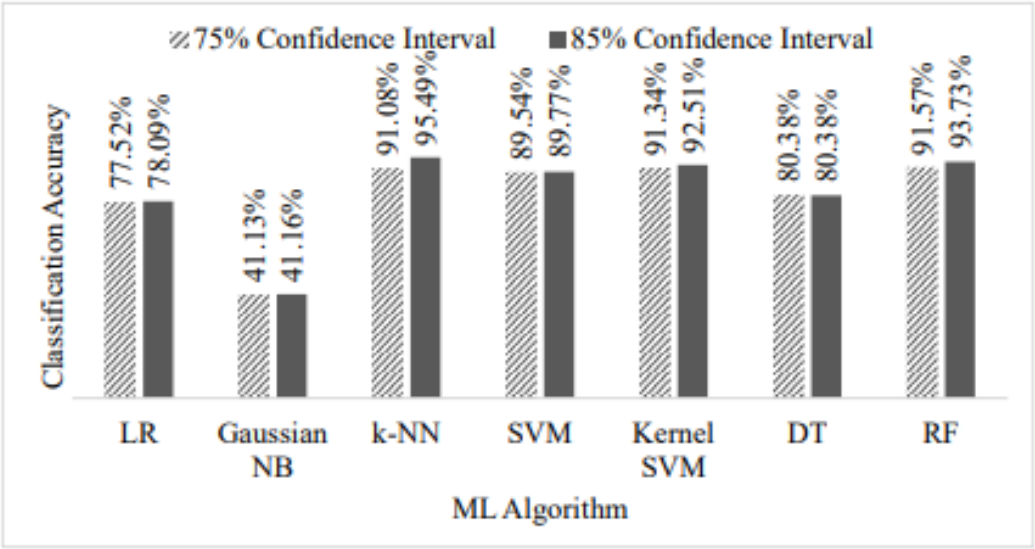


Figure 4.7 SCADA: Scenarios Overall Accuracy Classification, One or Two Scenario(s) Based on 75% and 85% Confidence Intervals (5-fold cross-validation)

The next part analyses and evaluates the performance of identifying abnormalities in IoT networks using the same ML algorithms. An innovative dataset based on MQTT is created and put to the test.

4.3 MQTT IDS DATASET GENERATION

With their widespread adoption over the last decade, IoT devices are projected to reach 25.1 billion by 2025. Many industries and fields make use of IoT networks; such examples are

smart cities, agriculture, healthcare, and supply chain management. Among the many protocols that allow for communication between machines in the Internet of Things, MQTT stands out. With an emphasis on the MQTT protocol, Harsha et al. examine the protocols used in IoT networks. The authors' study emphasizes the necessity for specialized intrusion detection systems by identifying the many security vulnerabilities connected with MQTT. Their research reveals that 53396 MQTT devices are accessible to the public. The security risks of MQTT are further examined by Dinculeana and Cheng. Based on their findings, more effective methods of detecting MQTT assaults are required.

There is an immediate need to provide current IoT datasets for IDS use since there is a dearth of statistics that reflect IoT traffic. According to the findings and the author's understanding, no intrusion detection system datasets are currently accessible that include MQTT communication, whether it be benign or malicious.

Additionally, an IoT intrusion detection system (IDS) is required due to the growing reliance on IoT and the fact that existing general-purpose IDSs are inadequate for meeting IoT requirements. However, processing, training, and evaluating classification models need datasets that are readily available. Here we provide an IDS dataset that was created using MQTT. First of its kind, this dataset mimics a MQTT-based network with both benign and malicious traffic, standing in for both general and MQTT-specific cyberattacks.

4.3.1 MQTT-IoT-IDS2020

A simulated MQTT network architecture is used to build the "MQTT-IoT-IDS2020" dataset, which represents the communication of an IoT network. Twelve sensors, a broker, a virtual camera, and an adversary make up the network. A total of five scenarios, including one for regular operation and four for attacks, are documented. The assailant carries out the four assaults, and each one is documented separately.

The assault scenarios are designed to include both specific assaults that target IoT networks and more generalized ones that are known to target MQTT. There are seven steps to an assault on an Internet of Things (IoT) network, say Dietz et al. "Scan" is the first step, and "Brute-force" is the second. According to Hafeez et al.'s five-tiered model for Internet of Things (IoT) threat models, network scanning is the first tier. Therefore, the attack scenarios are:

• Aggressive scan (Scan A)

• UDP scan (Scan sU)

• Sparta SSH brute-force (Sparta)

• MQTT brute-force attack (MQTT BF)

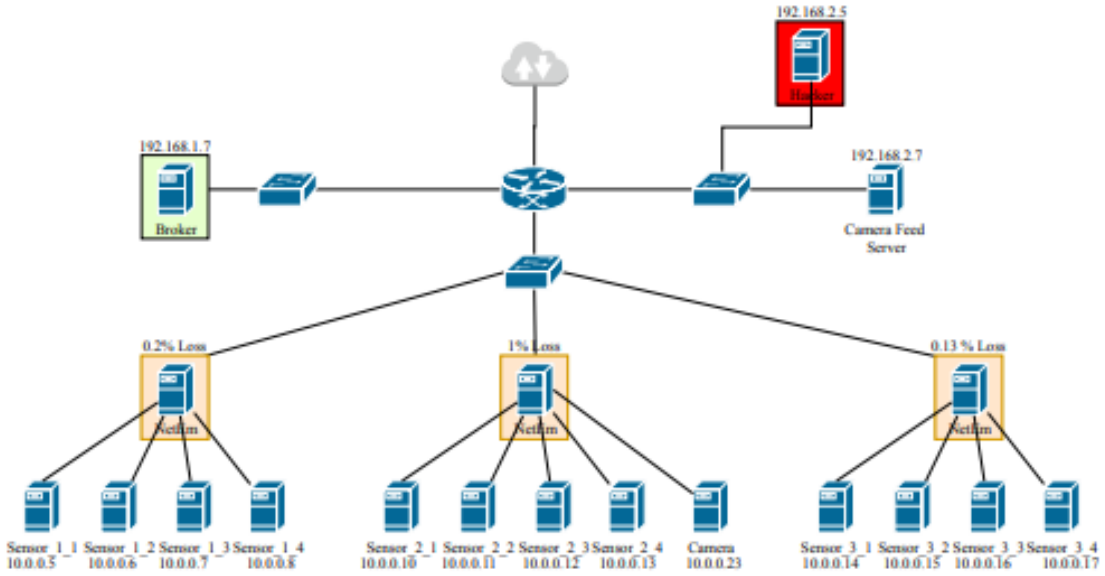


Figure 4.8 MQTT-IoT-IDS2020: Network Architecture

You can see the parts of the network in Figure 4.8. As it receives communications from all subscribers, the broker is responsible for delivering them and directing them to their final destinations. Separated into three subnetowrks, the twelve sensors represent three distinct parts. When everything is running well, each of the twelve sensors uses the "Publish" MQTT command to transmit a randomly generated message. The content is produced at random, and the message length changes between sensors to mimic various use cases. A continuous UDP stream is sent to the camera feed server using VLC media player to imitate a camera feed. The Ethernet ports on the router are used to record the network data. Additionally, in order to mimic real-world scenarios, the network discards packets. The average acceptable loss rates were used to determine the dropping rates, which are 0.2 percent, 1 percent, and 0.13%. First, the benign traffic, also known as normal traffic, is recorded. After then, as several assaults are being recorded, the operation proceeds in the background as traffic. The IP address of the attacker, "192.168.2.5," is shown in Figure 4.8 and is subsequently utilized for labeling the cases in the dataset.

To record data transfer across a network, one may use tcpdump. This is how the following instruments are employed:

• Virtual machines are used to simulate the network devices.

• Nmap is used for the scanning attacks.

• MQTT-PWN is used for the MQTT brute-force attack.

The OS specification is as follows:

• Sensors: Tiny Core Linux

• Camera and camera feed server: Ubuntu

• Hacker: Kali Linux

A PCAP file is stored in its raw form for every scenario. Three levels of abstraction characteristics are extracted from the five files: normal.pcap, sparta.pcap, scan A.pcap, mqtt bruteforce.pcap, and scan sU.pcap. I. Packet features; II. Unidirectional flow features; and III. Bidirectional flow features are the degrees of features. Features based on flows depict the data transfer between network nodes; they include metrics like the average amount of time between packets and the total number of packets in a flow. Files suitable for machine learning (CSV) are used to store the features. In order to facilitate more research on MQTT network traffic and related threats, the raw PCAP files are made publicly accessible with open-access rights. The three levels of feature abstraction are employed entirely, as mentioned in Section 4.6.

For a summary of the characteristics retrieved from the original PCAP files, see Table 4.7. Features based on packets are shown in the fourth column. Column six displays characteristics of bidirectional flow, whereas column five displays characteristics of unidirectional flow. There are characteristics (marked with an asterisk) that have two values for bidirectional flows: one for the forward flow and one for the backward flow. The prefixes "fwd" (for forward) and "bwd" (for backward) are used to record and differentiate between the two characteristics. Table 4.8 lists the distribution of cases.

Table 4.7 MQTT-IoT-IDS2020: Feature List and Description

Feature	Variable	Data Type	Packet	Uni-flow	Bi-flow
Source IP Address	ip_src	Text	✓	✓	✓
Destination IP Address	ip_dest	Text	✓	✓	✓
Last layer protocol	protocol	Text	✓		
Time to live	ttl	Integer	✓		
Packet Length	ip_len	Integer	✓		
Don't fragment IP flag	ip_flag_df	Binary	✓		
More fragments IP flag	ip_flag_mf	Binary	✓		
Reserved IP flag	ip_flag_rb	Binary	✓		
Source Port	prt_src	Integer	✓	✓	✓
Destination Port	prt_dst	Integer	✓	✓	✓
Transport Layer protocol (TCP/UDP)	proto	Integer		✓	✓
Reserved TCP flag	tcp_flag_res	Binary	✓		
Nonce sum TCP flag	tcp_flag_ns	Binary	✓		
Congestion Window Reduced TCP flag	tcp_flag_cwr	Binary	✓		
ECN Echo TCP flag	tcp_flag_ecn	Binary	✓		
Urgent TCP flag	tcp_flag_urg	Binary	✓		
Acknowledgement TCP flag	tcp_flag_ack	Binary	✓		
Push TCP flag	tcp_flag_push	Binary	✓		
Reset TCP flag	tcp_flag_reset	Binary	✓		
Synchronization TCP flag	tcp_flag_syn	Binary	✓		
Finish TCP flag	tcp_flag_fin	Binary	✓		
Number of Packets in the flow	num_pkts	Integer		✓	*
Average inter arrival time	mean_iat	Decimal		✓	*
Standard deviation of inter arrival time	std_iat	Decimal		✓	*
Minimum inter arrival time	min_iat	Decimal		✓	*
Maximum inter arrival time	max_iat	Decimal		✓	*
Number of bytes	num_bytes	Integer		✓	*
Number of push flag	num_psh_flags	Integer		✓	*
Number of reset flag	num_rst_flags	Integer		✓	*
Number of urgent flag	num_urg_flags	Integer		✓	*

Feature	Variable	Data Type	Packet	Uni-flow	Bi-flow
Average packet length	mean_pkt_len	Decimal		✓	*
Standard deviation packet length	std_pkt_len	Decimal		✓	*
Minimum packet length	min_pkt_len	Decimal		✓	*
Maximum packet length	max_pkt_len	Decimal		✓	*
MQTT message type	mqtt_messagetype	Integer	✓		
mqtt_messagelength MQTT message length		Binary	✓		
User Name MQTT Flag	mqtt_flag_uname	Binary	✓		
Password MQTT flag	mqtt_flag_passwd	Binary	✓		
Will retain MQTT flag	mqtt_flag_retain	Binary	✓		
Will QoS MQTT flag	mqtt_flag_qos	Integer	✓		
Will flag MQTT flag	mqtt_flag_willflag	Binary	✓		
Clean MQTT flag	mqtt_flag_clean	Binary	✓		
Reserved MQTT flag	mqtt_flag_reserved	Binary	✓		
Is Attack	is_attack	Binary (1 if the instance is attack, 0 otherwise)	✓	✓	✓

Table 4.8 MQTT-IoT-IDS2020: Instances Distribution

File Name	PCAP file size	Number of Packets		Number of Uni-flow Instances		Number of Uni-flow Instances	
		Benign	Attack	Benign	Attack	Benign	Attack
normal	192.5 MB	1056230 (3.42%)	0	171836 (59.01%)	0	86008 (54.78%)	0
scan_A (aggressive)	16.2 MB	70768	40624 (0.13%)	11560	39797 (13.67%)	5786	19907 (12.68%)
scan_sU (UDP)	41.3 MB	210819	22436 (0.07%)	34409	22436 (7.71%)	17230	22434 (14.29%)
sparta	3.4 GB	947177	19728943 (63.93%)	154175	28232 (9.7%)	77202	14116 (8.99%)
mqtt_bf	9.6 MB	32164	10013142 (32.45%)	4205	28874 (9.92%)	2152	14544 (9.26%)

4.4 MQTT

We use the six ML algorithms—LR, Gaussian NB, k-NN, SVM, DT, and RF—discussed previously to evaluate various ML approaches on the MQTT-IoT-IDS2020 dataset. The following characteristics are disabled to prevent any identifying data from being impacted:

protocol, MQTT flags, source and destination IP addresses. The data is divided into two parts: training and testing. Each experiment is evaluated using five-fold cross-validation. Along with each class's precision, recall, and F1-Score, the total accuracy is used for assessment. As a last step in analyzing the IDS performance, the weighted average (W. AVG.) of the recall, precision, and F1 score is computed.

Table 4.10 shows the total accuracy of all ML methods with packet, unidirectional, and bidirectional feature levels. Using flow-based features improves overall accuracy for all methods (except NB), as shown in Table 4.10. The improved ability of flow-based characteristics to distinguish between harmless and malicious MQTT-based assaults is the rationale for the noticeable improvement in accuracy (Figure 4.9).

Table 4.9 MQTT-IoT-IDS2020: Overall Detection Accuracy

	Features		
	Packet	**Unidirectional**	**Bidirectional**
LR	78.87%	98.23%	99.44%
k-NN	69.13%	99.68%	99.9%
DT	88.55%	99.96%	99.95%
RF	65.39%	99.98%	99.97%
SVM (RBF Kernel)	77.4%	97.96%	96.61%
NB	81.15%	78%	97.55%
SVM (Linear Kernel)	66.69%	82.6%	98.5%

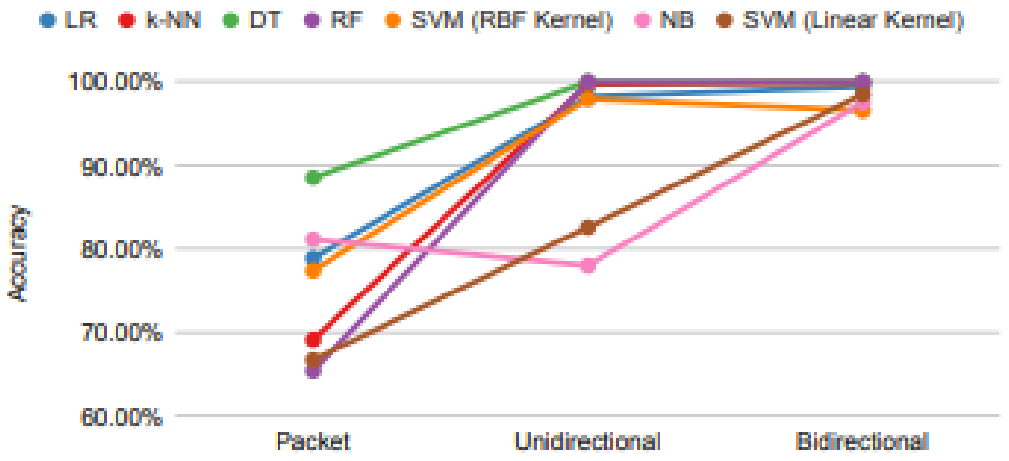


Figure 4.9 MQTT-IoT-IDS2020: Overall Detection Accuracy Trend using Different ML Techniques

Using flow-based features also yields the best overall accuracy for RF (see Table 4.10). utilizing bidirectional flow features resulted in a 99.97% accuracy rate, whereas utilizing unidirectional flow features obtained 99.96%. As these findings are the average of 5-fold cross validation, the 0.01% difference between mono and bidirectional flow characteristics is considered trivial. The findings, however, show that RF is successful here; this is because the approach is well-suited to multiclass issues and can handle several characteristics without overfitting. All five classes achieve F1-Scores, recalls, and precisions over 99.9%, as shown in Table 4.11.

Table 4.10 MQTT-IoT-IDS2020 Results: LR - k-NN - DT (5-fold cross-validation)

	LR								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	0%	100%	99.02%	0%	93.33%	98.95%	0%	96.55%	98.99%
Scan_A	86.45%	70.87%	97.25%	98.39%	98.39%	97.21%	92.03%	82.39%	97.2%
Scan_sU	98.21%	98.03%	98.48%	99.34%	95.76%	100%	98.77%	96.88%	99.23%
Sparta	100%	100%	100%	98.22%	100%	100%	99.1%	100%	100%
MQTT_BF	100%	99.25%	99.58%	51.75%	99.82%	99.41%	68.2%	99.53%	99.5%
W. AVG.	78.87%	98.23%	99.44%	70.4%	98.32%	99.44%	72.97%	98.14%	99.44%

	k-NN								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	17.43%	99.69%	99.95%	17.42%	98.85%	99.59%	17.43%	99.27%	99.77%
Scan_A	99.99%	99.97%	100%	99.99%	99.85%	99.9%	99.99%	99.91%	99.95%
Scan_sU	99.99%	99.96%	100%	99.99%	99.96%	100%	99.99%	99.96%	100%
Sparta	100%	100%	100%	100%	100%	100%	100%	100%	100%
MQTT_BF	25.84%	99.3%	99.75%	25.85%	99.82%	99.97%	25.84%	99.56%	99.86%
W. AVG.	69.13%	99.68%	99.9%	69.13%	99.68%	99.9%	69.13%	99.68%	99.9%

Table 4.11 MQTT-IoT-IDS2020 Results: DT - RF - SVM - NB (5-fold cross-validation)

	DT								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	69.29%	99.92%	99.88%	69.39%	99.92%	99.91%	69.34%	99.92%	99.9%
Scan_A	100%	100%	100%	99.98%	99.95%	99.9%	99.99%	99.97%	99.95%
Scan_sU	99.98%	99.91%	100%	100%	100%	100%	99.99%	99.96%	100%
Sparta	100%	100%	100%	100%	100%	100%	100%	100%	100%
MQTT_BF	72.56%	99.95%	99.93%	72.47%	99.95%	99.93%	72.51%	99.95%	99.93%
W. AVG.	88.55%	99.96%	99.95%	88.55%	99.96%	99.95%	88.54%	99.96%	99.95%

	RF								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	9.34%	99.96%	99.93%	8.99%	99.94%	99.95%	9.16%	99.95%	99.94%
Scan_A	100%	100%	100%	99.98%	99.95%	99.95%	99.99%	99.97%	99.98%
Scan_sU	99.98%	99.91%	99.96%	99.99%	100%	100%	99.99%	99.96%	99.98%
Sparta	100%	100%	100%	100%	100%	100%	100%	100%	100%
MQTT_BF	15.15%	99.96%	99.97%	15.69%	99.98%	99.96%	15.42%	99.97%	99.97%
W. AVG.	65.39%	99.98%	99.97%	65.44%	99.98%	99.97%	65.41%	99.98%	99.97%

	SVM (RBF Kernel)								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	30.23%	100%	100%	28.13%	92.67%	87.13%	28.8%	96.19%	93.12%
Scan_A	83.8%	70.16%	42.13%	99.99%	96.18%	99.88%	91.18%	81.13%	59.22%
Scan_sU	92.33%	99.96%	100%	99.74%	93.01%	94.34%	95.89%	96.36%	97.09%
Sparta	100%	100%	100%	91.17%	100%	100%	95.38%	100%	100%
MQTT_BF	72.42%	98.44%	98.3%	53.56%	100%	100%	59.53%	99.22%	99.14%
W. AVG.	77.4%	97.96%	96.61%	74.35%	98.05%	97.02%	74.89%	97.87%	96.15%

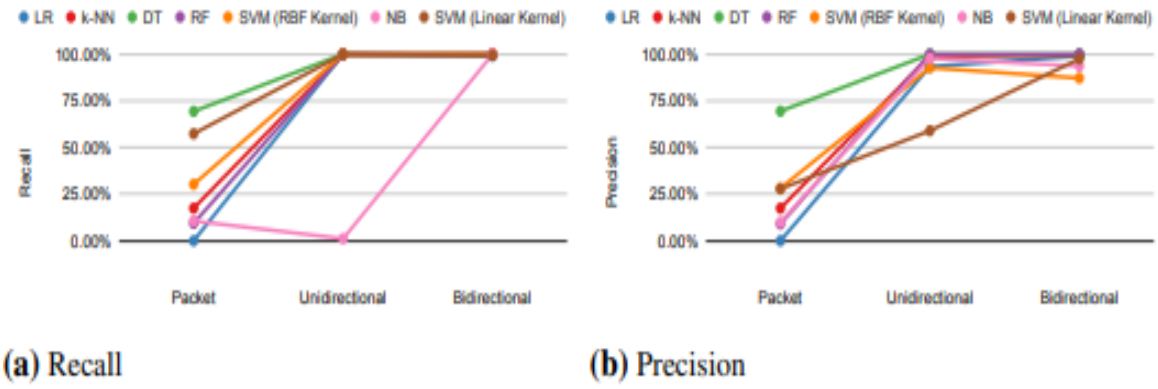
	SVM (Linear Kernel)								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	57.34%	99.84%	99.26%	27.8%	58.95%	97.45%	37.38%	73.82%	98.32%
Scan_A	83.28%	68.23%	84.1%	70.42%	70.35%	93.44%	69.7%	67.5%	87.01%
Scan_sU	78.13%	60.31%	97.76%	75.8%	70.71%	93.77%	76.92%	61.91%	95.27%
Sparta	87.64%	60.37%	99.99%	97.62%	99.94%	100%	89.89%	74.61%	99.99%
MQTT_BF	24.89%	97.79%	98.71%	43.3%	99.89%	99.55%	20.84%	98.83%	99.13%
W. AVG.	66.69%	82.6%	98.5%	65.42%	88.9%	98.66%	60.4%	82.42%	98.46%

	NB								
	Recall			Precision			F1-Score		
	Packet	Uni	Bi	Packet	Uni	Bi	Packet	Uni	Bi
Benign	10.62%	1.13%	99.96%	9.9%	97.68%	93.56%	10.25%	2.24%	96.65%
Scan_A	100%	99.25%	66.41%	99.23%	18.28%	100%	99.61%	30.88%	79.81%
Scan_sU	99.52%	97.76%	100%	100%	98.79%	98.52%	99.76%	98.27%	99.25%
Sparta	99.84%	100%	100%	100%	100%	100%	99.92%	100%	100%
MQTT_BF	90.27%	97.78%	100%	53.15%	100%	97.05%	65.84%	98.88%	98.5%
W. AVG.	81.15%	78%	97.55%	73.29%	95.43%	98.37%	75.99%	75.26%	97.77%

Tables 4.10 and 4.11 provide the accuracy, recall, and F1-Score for every classifier, and by comparing them to the flow-based characteristics, we can see how important these features are. According to Table 4.9, the performance is improved when flow-based features are used.

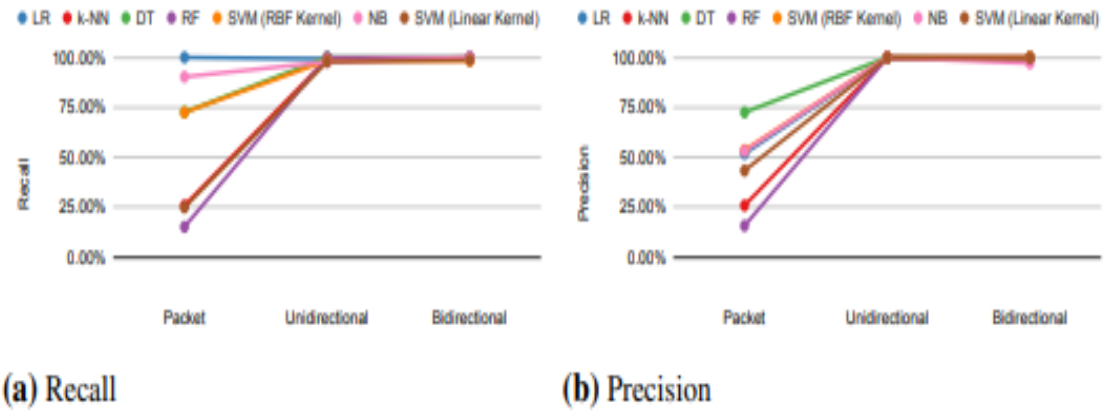
Both the benign and the MQTT-BF attack classes see a considerable improvement in performance when flow-based features are applied. When contrasted with networks designed for other purposes, benign operation traffic in IoT networks is rather straightforward. The needs and use of the IoT network will determine this. Thus, it is noticeable when an attacker launches a generic network-based assault, such as scanning. Attacks based on MQTT, on the other hand, are difficult to foil as they depend on the well-known instructions for MQTT communication. This means that regardless of the ML method employed, packet-based characteristics cannot distinguish between benign and malicious MQTT BF attacks.

When it comes to benign class recall, the NB classifier clearly performs poorly. The reason for this is because this classifier uses a conditional probabilistic method, which means that both packet-based features and unidirectional flow characteristics do not discriminate against innocuous traffic. Given these characteristics, it is impossible to determine the conditional likelihood of regular operation vs a MQTT BF assault. The performance trend charts show this behavior. Figure 4.10 shows the improvement in the accuracy and recall of benign traffic, Figure 4.11 shows the MQTT BF attack's recall and accuracy, and Figure 4.12 shows the overall weighted average recall and precision, both of which reflect this increase.



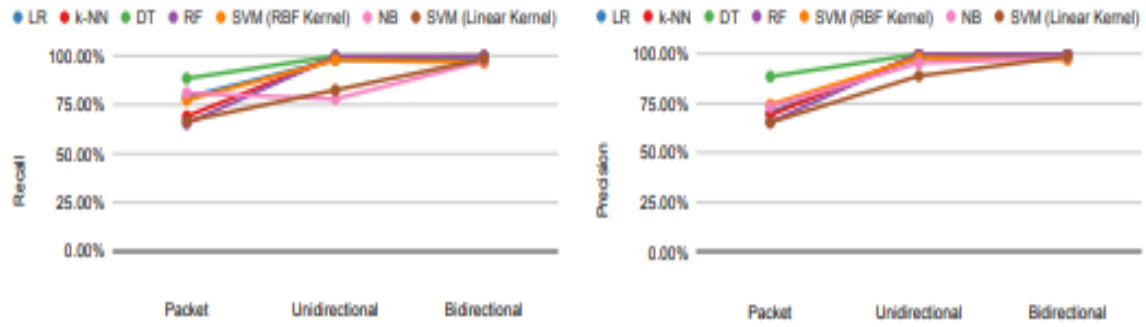
(a) Recall (b) Precision

Figure 4.10 MQTT-IoT-IDS2020: Benign Class Performance Trends



(a) Recall (b) Precision

Figure 4.11 MQTT-IoT-IDS2020: MQTT BF Class Performance Trends



(a) Recall (b) Precision

Figure 4.12 MQTT-IoT-IDS2020: Weighted Average Trends

The accuracy of the network's classification based on learning similarities using small samples from each class is specified by the assessment. Input (bold), output (italic), and hyperparameters (learning rate, ANN design, number of hidden layers, etc.) for the twin networks are as follows (Dr: Dropout layer, input: Siamese network, etc.)

• Twin Networks Architecture:

– SCADA: In(10) : 8

– CICIDS2017: In(31) : 25 : Dr(0.1) : 20 : Dr(0.05) : 15

– NSL-KDD - KDD Cup'99:

In(118) : 98 : Dr(0.1) : 79 : Dr(0.1) : 59 : Dr(0.1) : 39 : Dr(0.1) : 20

• Activation: Relu

• L2 regularisation: 0.001

• Optimiser: Adam

• Number of epochs: 2000

• Loss: Constructive loss

4.5 SCADA Dataset Results

Table 4.12 Sample Confusion Matrix

	Predicted Class				
Correct	Normal	$Attack_1$	$Attack_2$	$Attack_3$	$Attack_4$
Normal	TN	FP_1	FP_2	FP_3	FP_4
$Attack_1$	FN_1	TP_{11}	TP_{12}	TP_{13}	TP_{14}
$Attack_2$	FN_2	TP_{21}	TP_{22}	TP_{23}	TP_{24}
$Attack_3$	FN_3	TP_{31}	TP_{32}	TP_{33}	TP_{34}
$Attack_4$	FN_4	TP_{41}	TP_{42}	TP_{43}	TP_{44}

With j = 5, the total accuracy is 76.06%, as seen in Table 4.13. Nonetheless, the classes exhibit either a very high classification accuracy (nearing 100%) or a very poor accuracy (below 50%). Table 4.14 shows the model accuracy when various j-couples are used for voting. Note that simply labeling all occurrences as the majority class, a ZeroR (baseline majority classifier) will achieve a classification accuracy of 7.14%. Notably, the dataset just includes the register readings from the PLC, and the classes in the raw input space are tightly packed.

Table 4.13 Siamese Network: SCADA Classification Confusion Matrix (j = 5)

	Predicted Class														
Correct	Normal (S1)	Blocked measure 1 (S2)	Blocked measure 2 (S3)	DoS (S4)	Humidity (S5)	2 Floating objects (S6)	7 Floating objects (S7)	Person hitting high intensity (S8)	Person hitting med intensity (S9)	Person hitting low intensity (S10)	Plastic bag (S11)	Sensor failure (S12)	Spoofing (S13)	Wrong connection (S14)	Overall
S1	969 **(48.45%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	4 (0.2%)	110 (5.5%)	294 (14.7%)	0 (0%)	0 (0%)	376 (18.8%)	172 (8.6%)	0 (0%)	75 (3.75%)	
S2	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	
S3	0 (0%)	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	
S4	0 (0%)	0 (0%)	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	76.06%
S5	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	
S6	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2 (0.1%)	1416 **(70.8%)**	6 (0.3%)	152 (7.6%)	0 (0%)	0 (0%)	111 (5.55%)	1 (0.05%)	312 (15.6%)	0 (0%)	
S7	91 (4.55%)	469 (23.45%)	477 (23.85%)	0 (0%)	0 (0%)	5 (0.25%)	628 **(31.4%)**	0 (0%)	0 (0%)	0 (0%)	287 (14.35%)	24 (1.2%)	1 (0.05%)	18 (0.9%)	
S8	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	
S9	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	
S10	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	0 (0%)	2000 **(100%)**	0 (0%)	0 (0%)	0 (0%)	0 (0%)	
S11	312 (15.6%)	0 (0%)	0 (0%)	0 (0%)	3 (0.15%)	96 (4.8%)	122 (6.1%)	43 (2.15%)	0 (0%)	0 (0%)	1279 **(63.95%)**	122 (6.1%)	20 (1%)	3 (0.15%)	
S12	306 (15.3%)	0 (0%)	0 (0%)	31 (1.55%)	0 (0%)	149 (7.45%)	60 (3%)	486 (24.3%)	0 (0%)	0 (0%)	423 (21.15%)	337 **(16.85%)**	201 (10.05%)	7 (0.35%)	
S13	0 (0%)	0 (0%)	0 (0%)	0 (0%)	83 (4.15%)	447 (22.35%)	0 (0%)	9 (0.45%)	0 (0%)	0 (0%)	26 (1.3%)	29 (1.45%)	1406 **(70.3%)**	0 (0%)	
S14	115 (5.75%)	197 (9.85%)	0 (0%)	0 (0%)	1 (0.05%)	0 (0%)	29 (1.45%)	0 (0%)	328 (16.4%)	0 (0%)	55 (2.75%)	12 (0.6%)	0 (0%)	1263 **(63.15%)**	

Table 4.14

Siamese Network: SCADA Classification Accuracy Using Different j Votes

No Votes (j)	Overall Accuracy	Normal	
		TNR	FPR
1	72.23%	34.3%	65.7%
5	76.06%	48.45%	51.55%
10	77.77%	45.35%	54.65%
15	78.6%	46.65%	53.35%
20	79.18%	47.3%	52.7%
25	79.06%	46.75%	53.25%
30	79.21%	45.7%	54.3%

The use of classification confidence to achieve a greater scenario classification accuracy was motivated by the overlap that was shown in Section 4.4 via the variant classification accuracies. By fitting a k-NN model to the input space data, we can further illustrate that these classes overlap. We choose k=30 to compare it to the Siamese network where 30

pairings are used for majority voting. It should be mentioned that the Siamese network similarity determines the distance between the twin networks' outputs, while the k-NN model determines the distance between instances in their input space. The number of classes that were incorrectly labeled for instances of each class c is shown in Figure 5.8. For instance, the k-NN and Siamese networks both mislabeled "S6: 2 Floating Objects" occurrences as seven and three, respectively. Figure 5.8 shows that the Siamese network model has mastered a transformation that lessens class overlap, which explains why S5: Humidity and S8-S10: Person Hitting performed better than before.

Also, the classifications that don't overlap, like S2: Blocked Measures, S3: DoS, and S4: Other, have very good accuracy when it comes to classifying things. Lastly, the Siamese network may decrease the amount of misclassified classes caused by overlap for classes that overlap with more than seven additional classes. On the other hand, these classes were not accurately classified using only "pair similarity."

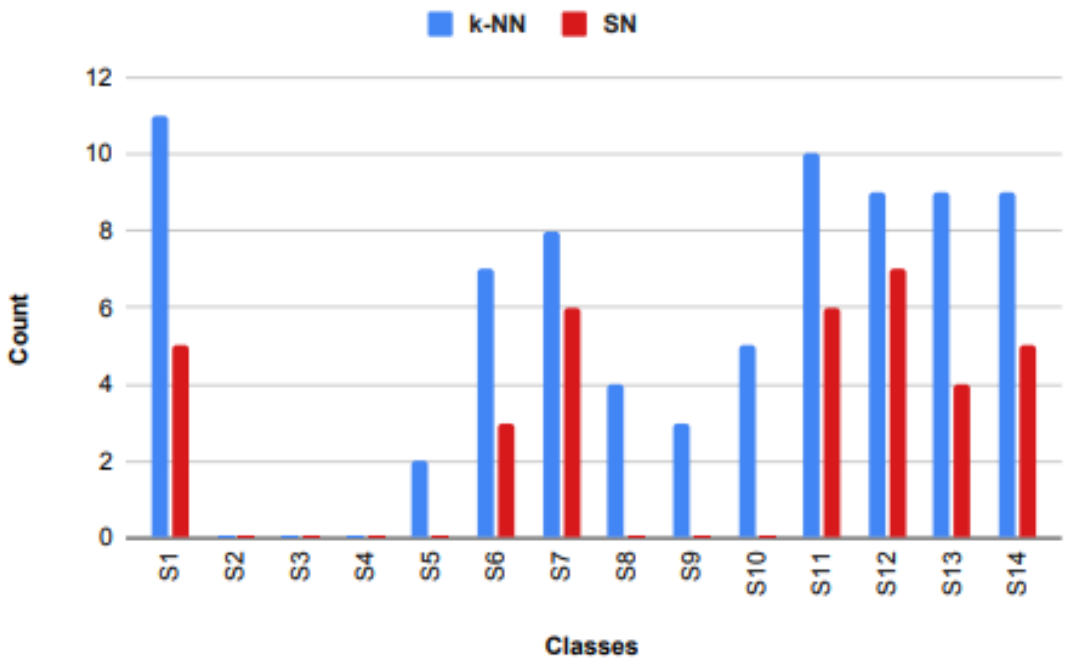


Figure 4.13 SCADA Dataset k-NN (k=30) and Siamese Network (SN) (30 pairs): Number of Wrong Associated Classes During Classification

4.6 CICIDS2017 Dataset Results

Table 4.15 displays the CICIDS2017 CM of the categorization. With j = 5, the total accuracy is 83.74% based on pairs similarity alone, as reported. The accuracy rates for the various attack classes are as follows: 96.08%, 75.17%, 80.05%, and 76.55%. In addition, the first column displays the low false negative rates. Additionally, the attack classes had minimal FPRs: 0.05% for Normal, 2.6% for Advanced, 1.87% for Advanced, and 4.62% for Advanced.

Table 4.15 Siamese Network: CICIDS2017 Classification Confusion Matrix (j = 5)

	Predicted Class					
Correct	Normal	DoS (Hulk)	DoS (Slowloris)	FTP	SSH	Overall
Normal	5452 **(90.87%)**	3 *(0.05%)*	156 *(2.6%)*	112 *(1.87%)*	277 *(4.62%)*	83.74%
DoS (Hulk)	139 *(2.32%)*	5765 **(96.08%)**	24 *(0.4%)*	13 *(0.22%)*	59 *(0.98%)*	
DoS (Slowloris)	914 *(15.23%)*	1 *(0.02%)*	4510 **(75.17%)**	71 *(1.18%)*	504 *(8.4%)*	
FTP	790 *(13.17%)*	2 *(0.03%)*	95 *(1.58%)*	4803 **(80.05%)**	310 *(5.17%)*	
SSH	973 *(16.22%)*	0 *(0%)*	227 *(3.78%)*	207 *(3.45%)*	4593 **(76.55%)**	

Table 4.16 Siamese Network: CICIDS2017 Classification Accuracy Using Different j Votes

No Votes (*j*)	Overall Accuracy	Normal	
		TNR	FPR
1	74.55%	70.43%	29.57%
5	83.74%	90.87%	9.13%
10	84.54%	92.58%	7.42%
15	84.63%	93.07%	6.93%
20	84.69%	93.55%	6.45%
25	84.69%	93.73%	6.27%
30	84.71%	93.85%	6.15%

Various j-pairs for voting are detailed in Table 4.16, along with their respective overall accuracy, TNR, and FPR. When compared to employing only one pair, the findings show that 5 pairs significantly improve overall accuracy (from 74.55% to 83.74%) and TNR (70.43% to 90.87%). Due to the unpredictability of the instance selection, the bad performance of 1 pair is the result. The chance of successful similarity-based categorization improves as j increases because the probability of finding a representable pair grows. The cases' variation and outliers also impact this random selection procedure.

4.7 KDD Cup'99 and NSL-KDD Datasets Results

Table 4.17 displays the CM of the classification for the KDD Cup'99 dataset. There is a tiny percentage of attack classes that are incorrectly categorized as normal (0.1%, 0.97%, 0.27%, and 8% for the attack classes accordingly), but generally, the accuracy is 87.99% when j = 5. Table 4.18 shows that, as in the CICIDS2017 dataset, utilizing 5 pairings improves accuracy and TNR.

Table 4.17 Siamese Network: KDD Cup'99 Classification Confusion Matrix (j = 5)

	Predicted Class					
Correct	Normal	DoS	Probe	R2L	U2R	Overall
Normal	4423 **(73.72%)**	9 *(0.15%)*	492 *(8.2%)*	979 *(16.32%)*	97 *(1.62%)*	87.99%
DoS	6 *(0.1%)*	5920 **(98.67%)**	64 *(1.07%)*	10 *(0.17%)*	0 *(0%)*	
Probe	58 *(0.97%)*	254 *(4.23%)*	5453 **(90.88%)**	222 *(3.7%)*	13 *(0.22%)*	
R2L	16 *(0.27%)*	0 *(0%)*	39 *(0.65%)*	5786 **(96.43%)**	159 *(2.65%)*	
U2R	480 *(8%)*	0 *(0%)*	685 *(11.42%)*	21 *(0.35%)*	4814 **(80.23%)**	

Table 4.18 Siamese Network: KDD Cup'99 Classification Accuracy Using Different j Votes

No Votes (j)	Overall Accuracy	Normal	
		TNR	FPR
1	82.03%	69.27%	30.73%
5	87.99%	73.72%	26.28%
10	88.26%	73.67%	26.33%
15	88.29%	73.63%	26.37%
20	88.26%	73.65%	26.35%
25	88.23%	73.6%	26.4%
30	88.24%	73.6%	26.4%

Classification accuracy was somewhat enhanced when the Siamese network model was trained on the NSL-KDD dataset, an enhanced version of the KDD Cup'99 dataset that has had duplicates eliminated. Tables 4.19 and 4.20 show the CM and various j-vote performances, respectively, for the NSL-KDD dataset. As a whole, the accuracy improved to 91.01% from 87.99% in the KDD Cup'99 dataset.

Table 4.19 Siamese Network: NSL-KDD Classification Confusion Matrix (j = 5)

	Predicted Class					
Correct	Normal	DoS	Probe	R2L	U2R	Overall
Normal	5187 **(86.45%)**	47 *(0.78%)*	300 *(5%)*	315 *(5.25%)*	151 *(2.52%)*	91.01%
DoS	144 *(2.4%)*	5621 **(93.68%)**	217 *(3.62%)*	16 *(0.27%)*	2 *(0.03%)*	
Probe	159 *(2.65%)*	643 *(10.72%)*	5133 **(85.55%)**	44 *(0.73%)*	21 *(0.35%)*	
R2L	227 *(3.78%)*	0 *(0%)*	31 *(0.52%)*	5669 **(94.48%)**	73 *(1.22%)*	
U2R	214 *(3.57%)*	0 *(0%)*	92 *(1.53%)*	2 *(0.03%)*	5692 **(94.87%)**	

Table 4.20 Siamese Network: NSL-KDD Classification Accuracy Using Different j Votes

No Votes (j)	Overall Accuracy	Normal	
		TNR	FPR
1	86.61%	80.47%	19.53%
5	91.01%	86.45%	13.55%
10	91.1%	86.45%	13.55%
15	91.17%	86.4%	13.6%
20	91.24%	86.47%	13.53%
25	91.26%	86.42%	13.58%
30	91.3%	86.53%	13.47%

The little improvement may be explained by the Siamese network's learning strategy when comparing the NSL-KDD results to those from the KDD Cup'99 dataset. The Siamese network is able to avoid balance or duplicating problems since it learns from similarities instead of particular class attributes. This is also resolved by making sure there is a balanced representation of class pairings and randomly selecting the training batch pairs.

As far as the author is aware, no published studies have addressed intrusion detection systems (IDS) using Siamese networks or using similarity-based One-Shot learning. Below, however, we detail the results of recent studies that make use of such databases. Even while it's not easy to compare the models' performance to that of newer IDS models, the findings from each class are easier to understand thanks to their work.

Table 4.21 presents the assessment of recent works published in IDS. These researches highlight specific class metrics in addition to overall accuracy, with an emphasis on multi-class attack categorization. Table 4.21 clearly shows that total classification accuracy is greater than individual class performance, which is an essential point to remember. We may thank the issue of class inequality for this. In the CICIDS2017 dataset, for instance, the overall accuracy is 96%, with a TPR of 0% for SSH attacks and 3.1% for FTP attacks. Similar to how the overall accuracy is 92.6% in the KDD Cup'99 dataset, the TPR for the R2L is 24.3% and for the U2R it is 15.5%. The degree to which a dataset is imbalanced, the size of the dataset as a whole, and the complexity of the data all contribute to the

prevalence of the class imbalance issue. Size adjustments, resampling, and adjusting the impact of incorrectly categorizing under-and over-sampled classes on overall accuracy are typical methods for resolving class imbalance. The articles in Table 4.21 did not apply any of these strategies, which led to deceptive overall accuracy figures and a gap between classes detection accuracy and overall accuracy. Take notice that the issue of class imbalance was not a hurdle for the assessment given in this Chapter. This is because a pool of instances is used to randomly choose an equal number of pairings for training and testing, ensuring that there is balance.

Using the Siamese network model, the overall accuracy of KDD Cup'99 achieves 88%, as seen in this section and Table 4.21. This is in comparison to 92.6% in [176] and 99.8% in [182]. It has been shown that the Siamese network has a higher TPR for the attack classes when looking at the TPR of the various classes. As an example, in [182], the TPR of DoS, Probe, R2L, and U2R is 99.67%; in [176], it is 93.9%; and in comparison to [182], it is 98.9%, 96.9%, and 75%. In Siamese network model terms, the corresponding TPRs are 90.88%, 96.43%, 80.23%, and 24.91%. Similarly, the Siamese network model achieves an NSL-KDD overall accuracy of 91%, which is higher than the 77.8% reported in [176] and the 83.83% reported in [293]. Using the Siamese Network, the TPR of DoS, Probe, R2L, and U2R are 93.68%, 85.55%, 94.48%, and 94.87%, respectively, compared to 97.42%, 96.51%, 68.53%, and 95.14% in [177] and 86.63%, 83.73%, 35.15%, and 23.5% in [294]. Lastly, when compared to 96% in [184], the overall accuracy of CICIDS2017 utilizing the Siamese network model is 84%. In comparison to 98% and 77% in [184] and 0% and 3.1% in [176], the TPR of FTP and SSH classes using the Siamese network model is 80.05% and 76.55%, respectively.

Table 4.21 Recent IDS Studies for Multi-Class Classification Performance

Year/ Reference	ML Technique	Metric	Result
CICIDS2017			
2019 / [176]	Deep Neural Network with 1 Layer	Accuracy	Overall: 96%
		TPR	Normal: 64.6% SSH: 0% FTP:3.1% DDoS: 9.5%
2020 / [184]	Multi-layer Perceptron	Recall	SSH: 98% FTP: 77%
KDD Cup'99			
2019 / [176]	Deep Neural Network with 1 Layer	Accuracy	Overall: 92.6%
		TPR	Normal: 99.4% DoS: 93.9% Probe: 73.2% R2L: 24.3% U2R: 15.5%
2020 / [182]	Ensemble Model	Accuracy	Overall: 99.8%
		TPR	Normal: 99.99% DoS: 99.99% Probe: 98.9% R2L: 96.9% U2R: 75%
NSL-KDD			
2019 / [177]	Multi-layer Perceptron	Accuracy	Normal: 87.31% DoS: 97.42% Probe: 96.51% R2L: 68.53% U2R: 95.14%
2019 / [293]	Ensemble model	Overall	$KDDTest^{+}$: 83.83% $KDDTest^{-21}$: 78.33%
2019 / [176]	Deep Neural Network with 1 Layer	Accuracy	Overall: 77.8%
		TPR	Normal: 97.3% DoS: 77.7% Probe: 61% R2L: 43.3% U2R: 24.1%
2020 / [294]	Multi-CNN	Recall	$\underline{KDDTest^{+}}$: Normal: 91.19% DoS: 86.63% Probe: 83.73% R2L: 35.15% U2R: 23.50% $\underline{KDDTest^{-21}}$: Normal: 62.08% DoS: 77.04% Probe: 82.60% R2L: 35.15% U2R: 23.50%

CICIDS2017 Dataset

Autoencoder Results

This is the best possible design for the autoencoder model used with the CICIDS2017 dataset:

• ANN Architecture: In(18):Hidden(15):Hidden(9):Hidden(15):Out(18)

• Batch size: 1024

• L2 regularisation: 0.0001

• Number of epochs: 50

• Loss: L2 norm (Mean Square Error)

The accuracy of the autoencoder model for each CICIDS2017 class is listed in Table 4.22 The accuracy is defined differently for the benign and assault classes, so keep that in mind. Attack detection is a measure of the model's accuracy. If the reconstruction error is more than the specified threshold, then this will work. For the innocuous class, accuracy does not reflect assaults but rather the percentage of cases that are not designated as zero-day attacks, meaning that the reconstruction error is lower than the specified threshold.

Table 4.22 Zero-Day Detection: CICIDS2017 Autoencoder Results

Class	Detection Accuracy			
Threshold	0.2	0.15	0.1	0.05
Benign (Validation)	96.56%	95.19%	90.47%	81.13%
FTP Brute-force	5.18%	5.34%	6.73%	82.82%
SSH Brute-force	7.2%	8.38%	78.05%	80.51%
DoS (Slowloris)	65.63%	71.73%	78.13%	80.85%
DoS (GoldenEye)	66.98%	85.55%	87.71%	90.01%
DoS (Hulk)	98.23%	98.23%	98.34%	98.43%
DoS (SlowHTTPTest)	22.42%	24.03%	28.09%	39.02%
DDoS	83.47%	92.23%	97.88%	99.67%
Heartbleed	28.61%	28.9%	39.6%	43.64%
Web BF	9.7%	9.95%	82.04%	85.41%
Web XSS	11.14%	11.28%	96.38%	99.46%
Web SQL	16.67%	16.67%	22.22%	27.78%
Infiltration - Dropbox 1	47.06%	52.94%	94.12%	94.12%
Infiltration - Dropbox 2	85.71%	85.71%	100%	100%
Infiltration - Dropbox 3	16.3%	23.8%	89.5%	98.04%
Infiltration - Cooldisk	48.08%	51.92%	86.54%	92.31%
Botnet	17.46%	17.77%	37.15%	66.88%
PortScan	16.15%	28.37%	75.21%	98.47%

According to Table 4.22, the benign class accuracy is 96.56% with a 0.2 threshold, 95.19% with a 0.15 threshold, 90.47% with a 0.1 threshold, and 81.13% with a 0.05 threshold. In addition, there are three types of attack detection accuracy that may be seen. To begin, there are categories of cyberattacks that are clearly identifiable and different from benign ones. Examples of attacks whose detection accuracy is high over a range of thresholds [83% - 99%] include DoS (Hulk) and DDoS. Secondly, there are kinds of cyberattacks that are marginally different from benign ones; for instance, SSH Brute-force and Ports can. In this scenario, the detection accuracy is shown to be threshold dependent, with lower thresholds exhibiting an increase in accuracy. The impact of threshold value selection on detection accuracy is highlighted by this. In the third place, there are cyber-attack classes like Botnet, SQL Injection, and DoS-Slow HTTP Test that are recognized, but not with the same level of precision, since they are not identifiable from innocuous traffic. The malicious cyberattacks mimic the seemingly innocuous traffic patterns.

Figure 4.14 shows the various CICIDS2017 classes together with the detection accuracies for each threshold setting. Class I has very steady detection accuracy, class II has a sharp increase in detection accuracy in the right-most slice (0.05 threshold), and class III is indistinguishable from benign traffic; all three groups can be seen in Figure 4.14. At most, the benign accuracy (upper left) is 18.87% FPR, which is within an acceptable range with various criteria. These may be seen in further detail in Appendix F, which displays the ROC curves for all of the assault classes.

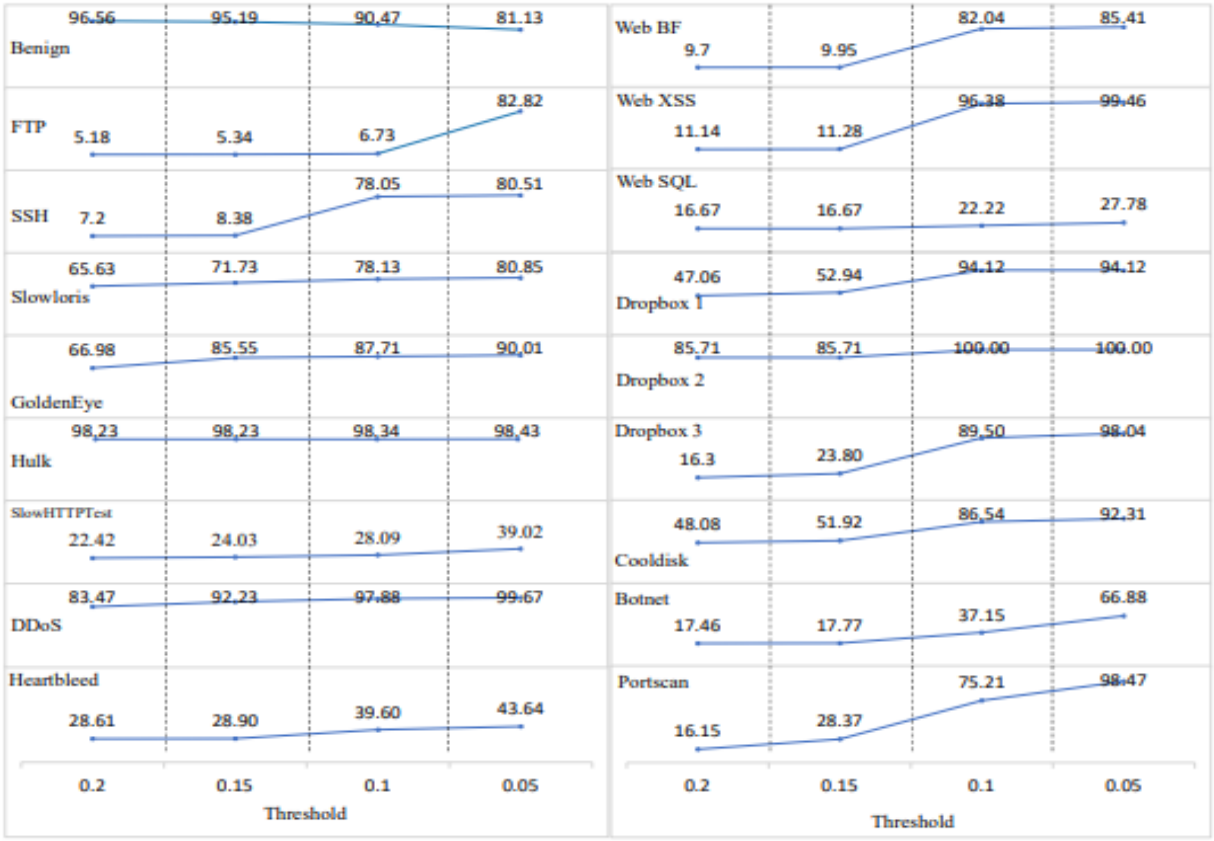


Figure 4.14 CICIDS2017 Autoencoder Detection Results Summary Per Class

One-Class SVM Results

Table 6.3 displays the results of the One-Class SVM model. Three things stand out from the analysis of the One-Class SVM results: (i) As ν grows, the benign detection accuracy drops. The accuracy of detection is 94.84% at ν = 0.05 and 79.71% at ν = 0.2. This is because there are more examples inside the decision boundary with a smaller ν in the One-Class SVM model. (ii) The One-Class SVM successfully identifies the classes with high detection accuracy in the autoencoder findings (Table 6.2), but it misses the other two

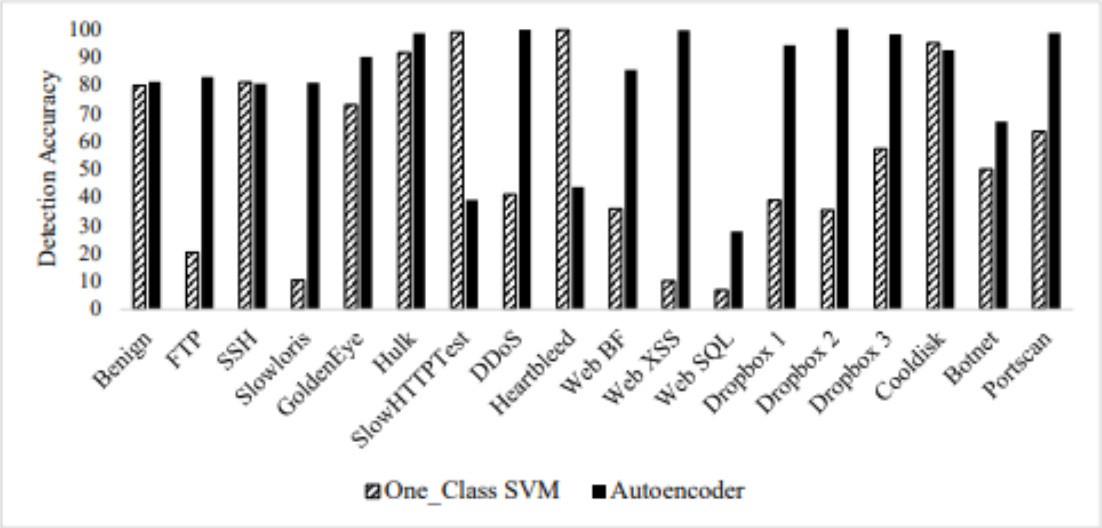
categories. The One-Class SVM method, which tries to fit a hyperplane to distinguish between benign and non-beneficial classes, has restrictions that cause this to happen. All classes that lie on this hyperplane will be considered normal or benign. Lastly, (iii) The detection percentage of the attack types that were accurately recognized ranges from 0% to 5%. To illustrate, the accuracy of DoS (Slow HTTP Test) detection is 98.11% at $\nu = 0.05$ and 98.71% at $\nu = 0.2$. Likewise, with $\xi = 0.05$, the SSH detection accuracy is 78.96%, and with $\nu = 0.2$, it increases to 80.95%.

Table 4.23 Zero-Day Detection: CICIDS2017 One-Class SVM Results

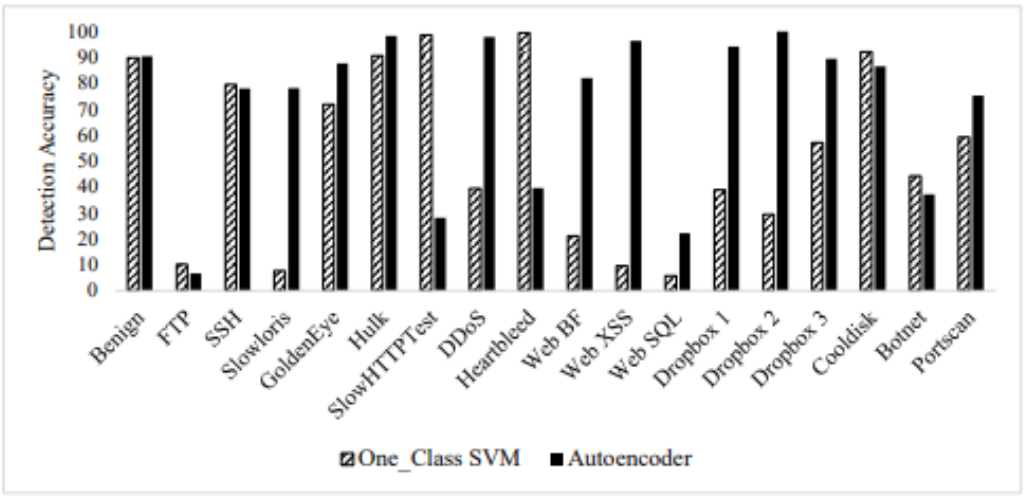
Class	Detection Accuracy			
ν	0.05	0.1	0.15	0.2
Benign (Validation)	94.84%	89.81%	84.84%	79.71%
FTP Brute-force	5.16%	10.19%	15.16%	20.29%
SSH Brute-force	78.96%	79.51%	80.26%	80.95%
DoS (Slowloris)	6.75%	7.66%	8.38%	10.37%
DoS (GoldenEye)	67.32%	71.87%	72.39%	72.85%
DoS (Hulk)	85.73%	90.69%	91.35%	91.55%
DoS (SlowHTTPTest)	98.11%	98.59%	98.66%	98.71%
DDoS	29.89%	39.35%	39.94%	40.96%
Heartbleed	99.09%	99.49%	99.54%	99.58%
Web BF	17.05%	21.1%	23.41%	35.84%
Web XSS	8.38%	9.58%	9.76%	10.13%
Web SQL	5.37%	5.77%	6.31%	6.85%
Infiltration - Dropbox 1	11.11%	38.89%	38.89%	38.89%
Infiltration - Dropbox 2	29.41%	29.41%	35.29%	35.29%
Infiltration - Dropbox 3	57.14%	57.14%	57.14%	57.14%
Infiltration - Cooldisk	90.96%	92.15%	93.8%	94.91%
Botnet	36.54%	44.23%	46.15%	50%
Portscan	57.61%	59.27%	60.04%	63.43%

Further visualisation of the contrast between the autoencoder and One-Class SVM models can be seen in Figure 4.15. When compared to the autoencoder model, One-Class SVM outperforms it in the DoS (Slow HTTP Test) and Heart bleed classes. Though their autoencoder reconstruction errors were lower than the zero-day criterion, the fact that these two classes were located on the other side of the One-Class SVM hyperplane explains how well they detected others. Hence, One-Class SVM works well for detecting known zero-day threats. Nevertheless, because to their much higher performance rating, autoencoders are more appropriate for sophisticated zero-day assaults. Figure 4.15 displays a comparison of the autoencoder and One-Class SVM performance on a class-by-class basis. Figure 4.15 (a) displays the outcomes obtained with an autoencoder threshold of 0.05 and One-Class

SVM ν = 0.2, while Figure 4.15 (b) displays the outcomes obtained with an autoencoder threshold of 0.1 and One-Class SVM ν = 0.09.



(a) SVM (ν = 0.2), AE (Threshold = 0.05)



(b) SVM (ν = 0.1), AE (Threshold = 0.1)

Figure 4.15 CICIDS2017 Autoencoder and One-Class SVM Comparison

4.8 Autoencoder Results for KDD Cup'99 and NSL-KDD Dataset

The autoencoder optimized architecture for the KDD Cup'99 and NSL-KDD datasets is:

• KDD Cup'99 ANN Architecture:

In(118):Hidden(100):Dr(0.2):Hidden(60):Dr(0.2):Hidden(100):Out(118)

• NSL-KDD ANN Architecture:

In(118):Hidden(122):Dr(0.2):Hidden(60):Dr(0.2):Hidden(100):Out(122)

• Batch size: 1024

• L2 regularisation: 0.001

• Number of epochs: 50

• Loss: L1 norm (Mean Absolute Error)

The KDD dataset family has opted for L1 (Mean Absolute Error) rather than L2 due to its superior performance. Additionally, L1 offered a more accurate measure for the reconstruction error because of the pre-engineered features and gave ranges of these two datasets. The autoencoder findings for the NSL-KDD dataset and the KDD Cup'99 dataset are listed in Table 4.24 and Table 4.25, respectively. The autoencoder is trained using 75% of the benign class, which is similar to the CICIDS2017 dataset. We test the model using attacks from the KDDTrain+ and KDDTest+ files for the NSL-KDD dataset. Random search parameter optimization is used to pick the threshold value, as previously stated. Although it is not as noteworthy as the debate around the CICIDS2017 dataset, the findings do show a trade-off between the threshold choice and the TNR. The reduced attack discrimination complexity and restricted attack coverage in the KDD datasets are to blame for this.

Table 4.24 Zero-Day Detection: KDD Cup'99 Autoencoder Results

Class	**Detection Accuracy**		
Threshold	0.3	0.25	0.2
Normal (Validation)	87.34%	83.95%	77.64%
DoS	99.4%	99.42%	99.48%
Probe	98.73%	98.93%	99.42%
R2L	96.36%	97.25%	100%
U2R	94.23%	96.15%	98.08%

Table 4.25 Zero-Day Detection: NSL-KDD Autoencoder Results

Class	Detection Accuracy		
Threshold	0.3	0.25	0.2
KDDTrain+.csv			
Normal (Validation)	79.09%	77.80%	72.78%
DoS	98.15%	98.16%	98.17%
Probe	99.89%	99.94%	99.94%
R2L	83.12%	96.48%	96.48%
U2R	84.62%	100%	100%
KDDTest+.csv			
Normal	84.82%	84.42%	80.94%
DoS	94.67%	94.67%	94.76%
Probe	100%	100%	100%
R2L	95.95%	96.5%	97%
U2R	83.78%	89.19%	100%

The findings shown here for the autoencoder perform better than the one currently available in the literature for identifying zero-day attacks. In their study, Gharib et al. use a hybrid two-stage autoencoder to distinguish between typical and non-standard traffic. While the authors claim an overall accuracy of 90.17% while training on the KDD Train+ file and testing on the KDD Test+, the suggested autoencoder in this section achieves 91.84%, 92.96%, and 94.54% with thresholds of 0.3, 0.25, and 0.2, respectively. Furthermore, it should be noted that Gharib et al. [178] refrain from providing specific data about their definition of zero-day attacks or the classes they use for testing. In this part, we build an autoencoder. Table 4.26 summarizes the performance comparison with Gharib et al. [178]. Additionally, it is shown that the autoencoder that was put into action works better than the one that was suggested for denoising in [163]. There was no mention of hyperparameter optimization or the elimination of associated characteristics in the paper. In addition, the occurrence of attacks affected their trials. To determine the appropriate threshold value, for instance, the authors trained an autoencoder using the attack cases. Additionally, attack cases were incorporated in the features normalization process. In particular, when it comes to KDD Test+ cases, this work's findings surpass the authors' model's accuracy maximum of 88%, which is 94%.

Table 4.26 Zero-Day Detection: NSL-KDD Performance Comparison

Year	Reference	Approach	Train:Test % of KDDTrain+	KDDTrain+ Accuracy	KDDTest+ Accuracy
This Work		AE th = 0.3 **AE th = 0.25** AE th = 0.2	75 : 25	88.97% **94.48%** 93.47%	91.84% 92.96% **94.54%**
2019	[178]	2 AEs	-	-	90.17%
2017	[163]	AE Denoising AE	80 : 20	93.62% 94.35%	88.28% 88.65%

One-Class SVM Results

Table 4.27 shows the results of the One-Class SVM for the NSL-KDD dataset while Table 4.28 shows the results for the KDD Cup'99 dataset. As mentioned in Section 4.5.2, the auto encoder's detection trends are comparable to these findings. This is because the KDD Cup'99 and NSL-KDD datasets only cover a small subset of possible assaults. Figure 4.16 shows the findings for the KDD Cup'99 dataset, whereas Figure 4.17 shows the results for the NSL-KDD dataset, allowing one to see the similarities in detection accuracy.

Table 4.27 Zero-Day Detection: KDD Cup'99 One-Class SVM Results

Class	**Detection Accuracy**		
ν	0.1	0.15	0.2
Normal (Validation)	90.15%	85.24%	79.93%
DoS	99.48%	99.49%	99.71%
Probe	99.05%	99.29%	99.37%
R2L	96.8%	97.51%	98.49%
U2R	96.15%	96.15%	98.08%

Table 4.28 Zero-Day Detection: NSL-KDD One-Class SVM Results

Class	Detection Accuracy		
ν	0.1	0.15	0.2
KDDTrain+.csv			
Normal (Validation)	89.9%	85.14%	80.54%
DoS	98.13%	98.14%	98.14%
Probe	97.74%	98.77%	99.52%
R2L	49.35%	52.26%	81.71%
U2R	78.85%	80.77%	82.69%
KDDTest+.csv			
Normal	88.12%	86.02%	84.72%
DoS	94.67%	94.67%	94.69%
Probe	99.55%	99.91%	100%
R2L	80.17%	82.22%	90.31%
U2R	78.38%	78.38%	83.78%

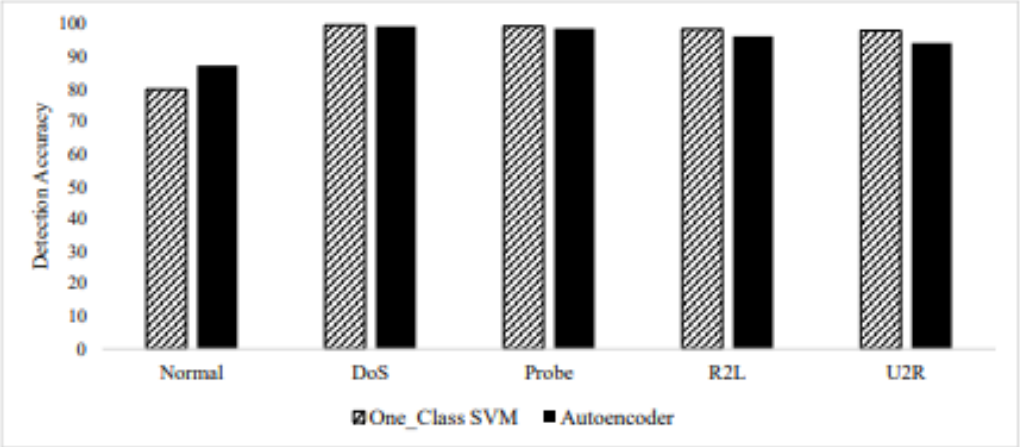


Figure 4.16 KDD Cup'99 Autoencoder and One-Class SVM Comparison SVM (v = 0.2), AE (Threshold = 0.3)

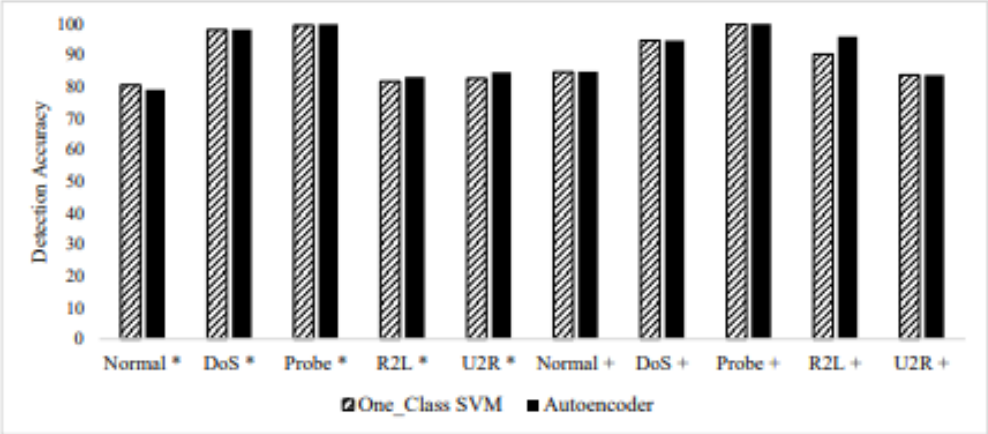


Figure 4.17 NSL-KDD Autoencoder and One-Class SVM Comparison SVM (v = 0.2), AE (Threshold = 0.3) *: KDDTrain+ file, +: KDDTest+ file

CHAPTER 5

LIMITATIONS OF THE STUDY

1) **Limited data Diversity**
   - The study relies on bench mark dataset such as KDD, NSL-KDD, CICIDS2017 and SCADA dataset, which may not fully represent real world SQL attack scenarios.
   - Lack of Large-Scale, real time SQL Query dataset limits generalization.

2) **Dependency on Simulated Environment**
   - IoT and SCADA experiments are conducted using simulated environments (e.g. MQTT-based datasets).
   - Real world deployment conditions (Noise, unpredictable traffic) are not fully captured.

3) **Limited SQL Injection Scope**
   - The focus is primarily on SQL Injection attacks, while other web-based attacks (XSS, CSRF) are not deeply analysed.
   - This restricts the system's application to bother web security threats.

4) **Model generalization Issue**
   - Some models perform well on specific dataset but may fail to generalize across different environments or datasets.
   - Overfitting risk exists, especially with structured datasets.

CHAPTER 6

CONCLUSION

Security of sensitive information, including financial and health records, remains a top concern due to SQL injection attacks and other web-based assaults. This problem is becoming more pressing as more and more social activities rely on the internet. We have demonstrated that the algorithms we have tested, including rule-based and decision tree algorithms, can classify testing data significantly faster and with accuracy comparable to that of Neural Networks, and we have also suggested a multi-source data analysis system to improve the accuracy of SQL injection attack detection. Adapting this system to detect other types of web-based attacks is on the list of future works. Other things on the list include gathering more data, like traffic going outbound from the web app to the browser, collecting larger datasets to see if that helps performance, and analyzing additional machine learning techniques for accuracy and performance.

Intrusion Detection Systems (IDS) are programs that scan every incoming and outgoing data packets for signs of malicious activity. In the last ten years, IDS have been built using a variety of ML approaches. The rapid emergence of new cyber threats has led to the widespread usage of ML approaches. This thesis delves into the use of ML approaches to construct IDS with specific purposes. In addition, this thesis explores the possibility of enhancing the efficiency and efficacy of next-generation IDS by using innovative DL approaches that have been successfully used in other areas of study.

Case studies are given here, one for an IoT network and one for a SCADA network, that demonstrate the use of ML approaches to construct specialized intrusion detection systems. An array of six ML methods—LR, NB, k-NN, SVM, DT, and RF—are used. One dataset mimics a control interface (CI) used in a water SCADA system. The fourteen real-life cases included in the collection address topics such as cyber assaults, sabotage, and hardware failures. We do three separate experiments. Binary classification (i.e., normal vs anomalous classes) is executed in the first experiment. The second test determines which part was impacted—none in the event of regular operation, and the part impacted by an anomaly or cyber assault—through classification. The third experiment seeks to categorize the situation that has occurred.

The three experiments showed that k-NN, DT, and RF outperformed NB, SVM, and LR. This is because the dataset properties are not linear, which is a result of the way CI networks are built. Commonplace and out-of-the-ordinary pursuits may coexist in such an environment. The accuracy rates were 95.49% for scenario classification, 82.71% for affected components, and 94.12% for binary classification. The scenario overlapping problem necessitates the use of a confidence interval to provide the most probable scenario (or two scenarios in the event that confidence is below the permitted interval).

The set of data is created and archived by means of an Internet of Things (IoT) network simulator that employs the widely-used MQTT protocol. In addition to regular operations, the MQTT-IoT-IDS2020 dataset also includes four cyber attack scenarios. There are three tiers of feature abstraction that are created after the dataset gathering phase: packet-based, unidirectional flow, and bidirectional flow.

Employing packet-based characteristics resulted in an overall accuracy of 88.55%, whereas employing unidirectional and bidirectional flow features yielded an accuracy of 99.98% and 99.9%, respectively. The analysis also showed that the innocuous class and the MQTT-Brute-Force attack class were the only ones whose performances were enhanced by applying flow-based features. Reason being, IoT traffic behavior differs from typical attacks like scanning in an IoT environment. On the other hand, attacks that depend on MQTT (protocol-based attacks) pose a problem since they interfere with regular traffic because they use well-known MQTT communication instructions.

We introduce a One-Shot learning model that can make do with sparse training data. With any luck, this will put an end to the inverse relationship between the data needs and the ML model size. With that goal in mind, this chapter presents a fresh approach to construct IDS. The Siamese networks that form the basis of this model do not rely on unique characteristics that distinguish across classes but rather on similarities between pairs. Aside from reducing training data requirements, learning from similarities also allows for the introduction and subsequent classification of novel cyber threats.

There are three cases when the Siamese network model is tested. Firstly, it assesses how well similarity-based learning works for intrusion detection systems. For this purpose, we evaluate Siamese networks' attack classification (i.e., discrimination) capabilities based just on pair similarity. The second scenario tests how well the suggested Siamese network

model can cope with novel cyber assaults without retraining on small samples. The third scenario concludes with an evaluation of the Siamese network model's ability to detect and label zero-day attacks in addition to known assaults (those included in the training set).

The SCADA, CICIDS2017, NSL-KDD, and KDD Cup'99 datasets are the ones that are utilized for assessment. The first dataset (SCADA) has a classification accuracy of 76.06% in the given situation. On the other hand, the classes' classification accuracy ranges from very poor (below 50%) to very high (over 100%). This is a result of the SCADA dataset's multi-label structure, which causes classes to overlap as they are not mutually exclusive. Using only similarity, the Siamese network is unable to distinguish between classes with a large degree of overlap. Classification findings show that, out of 14 classes in the dataset, the Siamese network can differentiate between no more than 7 classes that overlap.

The total accuracy rate for categorization in the CICIDS2017 is 83.74 percent. With accuracies of 96.08%, 75.17%, 80.05%, and 76.55%, the various assault types may be identified with relative ease. According to the findings, it is insufficient to use a single pair for assault classification. This is because, as the number of pairings grows, the chance of picking a pair that can be represented grows as well, because to the unpredictability of pair selection. This calls for the employment of many instances before the class label is determined using the majority vote mechanism. With an overall accuracy of 91.01% for the NSL-KDD dataset and 87.99% for the KDD Cup'99 dataset, similar behavior was seen.

prioritizes the development of a model that can identify zero-day assaults. The suggested approach makes use of autoencoders' encoding-decoding capabilities. The model is trained on benign traffic, and zero-day attacks are identified by depending on the reconstruction error. Also, a One-Class SVM is used to compare the performance of the proposed autoencoder.

The zero-day detection accuracy for DoS (GoldenEye), DoS (Hulk), PortScan, and DDoS assaults, respectively, achieves 90.01%, 98.43%, 98.47%, and 99.67% according to CICIDS 2017. Both the KDD Cup'99 and NSL-KDD datasets achieved detection accuracies of 95.21% and 92.96%, respectively.

The one-class support vector machine model demonstrates excellent performance in identifying zero-day attacks on the KDD Cup'99 and NSL-KDD datasets, as well as unique attack classes on the CICIDS2017 dataset. The autoencoder shows superior detection

accuracy when compared to One-Class SVM, which has been shown in the literature to be useful and highly accurate in novelty detection. The FPR is poor in both models. Lastly, the autoencoder and One-Class SVM models had reduced detection rates for the CICIDS2017 classes that imitate innocuous traffic behavior (Slow Loris, Slow HTTP Test). The reason for this is because these assaults are conducted with behavior that mimics innocuous traffic, which means that their reconstruction error is minimal.

CHAPTER 7

FUTURE SCOPE

1) **Extension to Multi-Attack Detection**
    1. Expand the proposed system beyond SQL Injection to detect:
        - Cross-site scripting (XSS)
        - Distributed Deniel of Service (DDos)
        - Cross-Site request Forgery(CSRF)
    2. Develop a unified web attack detection framework

2) **Real-time and Online Learning IDS**
    1. Design real-time IDS models for continuous monitoring of SQL queries.
    2. Implement online learning algorithms to update models dynamically without retraining.

3) **Advanced Deep learning Models**
    1. Integrate LSTM, GRU, transformer models for sequential SQL analysis.
    2. Apply graph Neural Networks (GNN) to model between queries and database entities.

4) **Zero-day Attack detection Enhancement**
    1. Improved detection of unknown and zero-day SQL attacks using:
        - Autoencoders
        - One-class SVM
        - Few-shot and One-Shot learning(e.g. Siamese Networks)

REFERENCE

[1] Hassan Azwar et.al “Intrusion Detection in secure network for Cybersecurity systems using Machine Learning and Data Mining” 2018 IEEE 5th International Conference on Engineering Technologies & Applied Sciences, 22- 23 Nov 2018,

[2] Singh, Garima & Kant, Dev & Gangwar, Unique & Singh, Akhilesh. (2015). SQL Injection Detection and Correction Using Machine Learning Techniques. 10.1007/978-3-319-13728-5_49.

[3] Yoon-Teck Bau et.al “Machine Learning Approaches to Intrusion Detection System Using BO-TPE” https://doi.org/10.2991/978-94-6463-094-7_9

[4] Alqahtani, Hamed & Sarker, Iqbal & Kalim, Asra & Hossain, Sohrab. (2020). Cyber Security Intrusion Detection using Machine Learning Techniques.

[5] Ali, Israr & Adil, Syed & Ebrahim, Mansoor. (2020). Intrusion Detection Framework for SQL Injection. 10.48550/arXiv.2009.13868.

[6] Aleesa, Ahmed & Bahaa, Bilal & Zaidan, A. & Sahar, Nan. (2020). Review of intrusion detection systems based on deep learning techniques: coherent taxonomy, challenges, motivations, recommendations, substantial analysis and future directions. Neural Computing and Applications. 32. 10.1007/s00521-019-04557-3.

[7] Ayeni Olaniyi A et.al “Intrusion Detection System using Deep Learning” 02 July 2024.

[8] Hadji, Abdelhakim & Raouyane, Brahim & Rachdi, Mohamed. (2023). SQL Injection Detection in Cloud Computing with Machine Learning Algorithms. 1-6. 10.1145/3607720.3607749.

[9] Fares, Ahmed & Sharawy, Mohamed & Zayed, Hala. (2011). Intrusion Detection: Supervised Machine Learning. journal of Computing Science and Engineering (JCSE). 5. 305-313. 10.5626/JCSE.2011.5.4.305.

[10] Zhou, Qianru & Li, Rongzhen & Xu, Lei & Zhu, Hongyi & Liu, Wanli. (2021). An Assessment of Intrusion Detection using Machine Learning on Traffic Statistical Data. 10.36227/techrxiv.14837994.

[11] Sohrabi, Mina & Javidi, M.M. & Hashemi, Sattar. (2013). Detecting intrusion transactions in database systems: A novel approach. Journal of Intelligent Information Systems. 42. 619-644. 10.1007/s10844-013-0286-z.

[12] Silva, Rui & Barbosa, Raul & Bernardino, Jorge. (2020). Intrusion Detection Systems for Mitigating SQL Injection Attacks: Review and State-of-Practice. International Journal of Information Security and Privacy. 14. 20-40. 10.4018/IJISP.2020040102.

[13] Kikissagbe, Brunel & Adda, Meddi. (2024). Machine Learning-Based Intrusion Detection Methods in IoT Systems: A Comprehensive Review. Electronics. 13. 3601. 10.3390/electronics13183601.

[14] Ross, Kevin & Moh, Melody & Moh, Teng-Sheng & Yao, Jason. (2018). Multi-source data analysis and evaluation of machine learning techniques for SQL injection detection. 1-8. 10.1145/3190645.3190670.

[15] Kamil, Ziadoon & Robiah, Y. & Bahaman, Nazrulazhar & Mostafa, Salama & Mohd Foozy, Cik Feresa. (2021). Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset. IEEE Access. 10.1109/ACCESS.2021.3056614.

[16] Khan, Amjad & Kashif, Muhammad & Jhaveri, Rutvij & Raut, Roshani & Saba, Tanzila & Bahaj, Saeed. (2022). Deep Learning for Intrusion Detection and Security of Internet of Things (IoT): Current Analysis, Challenges, and Possible Solutions. Security and Communication Networks. 2022. 1-13. 10.1155/2022/4016073.

[17] Wang, Y., Xi, J.S. and Cheng, T. (2021) The Overview of Database Security Threats' Solutions: Traditional and Machine Learning. Journal of Information Security, 12, 34-55. https://doi.org/10.4236/jis.2021.121002

[18] Hagar, Abdulnaser & Chaudhary, Deepali & Al-Bakhrani, Ali & Gawali, Dr. Bharti. (2021). Big Data Analytic Using Machine Learning Algorithms For Intrusion Detection System: A Survey. 10.24247/ijmperdjun2020575.

[19] Priyaa, B. & M, Indra Devi. (2016). Hybrid SQL injection detection system. 1-5. 10.1109/ICACCS.2016.7586332.

[20] Anthony, Cynthia & Elgenaidi, Walid & Rao, Muzaffar. (2024). Intrusion Detection System for Autonomous Vehicles Using Non-Tree Based Machine Learning Algorithms. Electronics. 13. 809. 10.3390/electronics13050809.

[21] Thakkar, Ankit & Lohiya, Ritika. (2023). A Review on Challenges and Future Research Directions for Machine Learning-Based Intrusion Detection System. Archives of Computational Methods in Engineering. 30. 10.1007/s11831-023-09943-8.

[22] Arjunan, Tamilselvan. (2024). Fraud Detection in NoSQL Database Systems using Advanced Machine Learning. International Journal of Innovative Science and Research Technology (IJISRT). 248-253. 10.38124/ijisrt/IJISRT24MAR127.

[23] Alsaeedi, Abdullah & Khan, Mohammad. (2019). Performance Analysis of Network Intrusion Detection System using Machine Learning. International Journal of Advanced Computer Science and Applications. 10. 671-678. 10.14569/IJACSA.2019.0101286.

[24] Lodeiro-Santiago, Moisés & Caballero-Gil, Cándido & Caballero-Gil, Pino. (2017). Collaborative SQL-injections detection system with machine learning. 1-5. 10.1145/3109761.3158395.

[25] Aljuaid, Wa'ad & Alshamrani, Sultan. (2024). A Deep Learning Approach for Intrusion Detection Systems in Cloud Computing Environments. Applied Sciences. 14. 5381. 10.3390/app14135381.

[26] Basma, Karbal & Rahal, Romadi. (2020). A Comparison of Different Machine Learning Algorithms for Intrusion Detection.

[27] Roua Dhahbi et.al "A Deep Learning Approach for Intrusion Detection"2022

[28] Chattopadhyay, Manojit & Sen, Rinku & Gupta, Sumeet. (2018). A Comprehensive Review and Meta-Analysis on Applications of Machine Learning Techniques in Intrusion Detection. Australasian Journal of Information Systems. 22. 10.3127/ajis.v22i0.1667.

[29] Jaw, E.; Wang, X. Feature Selection and Ensemble-Based Intrusion Detection System: An Efficient and Comprehensive Approach. Symmetry 2021, 13, 1764. https://doi.org/10.3390/sym13101764

[30] Othman, Suad & Ba-Alwi, Fadl & Nabeel, Taha & Al-Hashida, Amal. (2018). Intrusion detection model using machine learning algorithm on Big Data environment. Journal Of Big Data. 5. 10.1186/s40537-018-0145-4.

[31] Khraisat, Ansam & Gondal, Iqbal & Vamplew, Peter & Kamruzzaman, Joarder. (2019). Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity. 2. 10.1186/s42400-019-0038-7.

[32] Darwish, Saad. (2016). Machine Learning Approach to Detect Intruders in Database based on Hexplet Data Structure. Journal of Electrical Systems and Information Technology. 3. 10.1016/j.jesit.2015.12.001.

[33] Almseidin, Mohammad & Alzubi, Maen & Szilveszter, Kovács & Alkasassbeh, Mouhammd. (2018). Evaluation of Machine Learning Algorithms for Intrusion Detection System. 10.48550/arXiv.1801.02330.

[34] Note, Johan & Ali, Maaruf. (2022). Comparative Analysis of Intrusion Detection System Using Machine Learning and Deep Learning Algorithms. Annals of Emerging Technologies in Computing. 6. 19-36. 10.33166/AETiC.2022.03.003.

[35] Khraisat, A., Gondal, I., Vamplew, P. et al. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecur 2, 20 (2019). https://doi.org/10.1186/s42400-019-0038-7

[36] Doroudian, Mostafa & Shahriari, Hamid Reza. (2014). Database intrusion detection system for detecting malicious behaviors in transaction and inter-transaction levels. 2014 7th International Symposium on Telecommunications, IST 2014. 809-814. 10.1109/ISTEL.2014.7000815.

[37] Mustapha, Adeyinka & Udeh, Anayo & Ashi, Thomas & Sobowale, Odunayo & Akinwande, Mayowa & Oteniara, Aminat. (2024). Comprehensive review of machine learning models for sql injection detection in e-commerce. World Journal of Advanced Research and Reviews. 23. 10.30574/wjarr.2024.23.1.2004.

[38] Mohammed Kado Tufa “Developing Flow-Based Intrusion Detection Model for Software Defined Network Using Machine Learning Approach” February, 2022

[39] Surakhi, Ola & Mora, Antonio & Jamoos, Mohammed & .Alkhanafseh, Mohammed. (2022). The Intrusion Detection System by Deep Learning Methods: Issues and Challenges. The International Arab Journal of Information Technology. 19. 10.34028/iajit/19/3A/10.

[40] Sharafali, Shaimaa & Fallooh, Noor & Al-Hayani, Mohammed. (2023). Intrusion Detection System Based On Machine Learning and Deep Learning Techniques: A Review.

[41] Deepa Manikandan et.al “Machine learning approach for intrusion detection system using dimensionality reduction” Vol 34, No 1

[42] Roland Plaka et.al “Intrusion Detection Using Machine Learning For Industrial Control Systems” June 11, 2021

[43] Almutairi, Yasmeen & Alhazmi, Bader & Munshi, Amr. (2022). Network Intrusion Detection Using Machine Learning Techniques. Advances in Science and Technology Research Journal. 16. 193-206. 10.12913/22998624/149934.

[44] Singh, Amit & Prakash, Jay & Kumar, Gaurav & Jain, Praphula & Ambati, Loknath. (2024). Intrusion Detection System: Journal of Database Management. 35. 10.4018/JDM.338276.

[45] Tahsinur Rahman “Intrusion Detection system based on Deep Learning” July 29, 2022+

[46] Lakshit Sama “Network Intrusion Detection using Deep Learning” A thesis submitted for the degree of Masters of Research at Victoria University in July, 2022

[47] Kültür, Ebru “Network Intrusion Detection With A Deep Learning Approach” February 2022, 99 pages

[48] Kassahun Worku G/Michael” A Predictive Model Of Network Intrusion Detection Systems Using Machine Learning Approach” January, 2023

[49] Miss. Manali Raka et.al “Intrusion Detection using Machine Learning and log analysis” International Journal of Advanced Research in Computer and Communication Engineering Vol. 8, Issue 4, April 2019

[50] Vanin, P.; Newe, T.; Dhirani, L.L.; O’Connell, E.; O’Shea, D.; Lee, B.; Rao, M. A Study of Network Intrusion Detection Systems Using Artificial Intelligence/Machine Learning. Appl. Sci. 2022, 12, 11752. https://doi.org/10.3390/app122211752

[51] Adeyinka Ayodeji Mustapha et.al “Comprehensive review of machine learning models for sql injection detection in ecommerce” doi.org/10.30574/wjarr.2024.23.1.2004

[52] C. Gazala Akhtar et.al “SQL Injection Detection Using Machine Learning Techniques” Journal of critical Review ISSN -2394-5125 VOL 10, ISSUE 04,2023

[53] Gabriel C. Fernández “Deep Learning Approaches For Network Intrusion Detection”2019

[54] Mohammed Hamid Abdulraheem et.al “A Detailed Analysis Of New Intrusion Detection Dataset” Journal of Theoretical and Applied Information Technology 15th September 2019. Vol.97. No 17

[55] Hongyu Liu et.al “Machine Learning and Deep Learning Methods for Intrusion Detection Systems: A Survey” Appl. Sci. 2019, 9, 4396; doi:10.3390/app9204396

[56] Zhiyan Chen et.al “Machine learning-enabled hybrid intrusion detection system with host data transformation and an advanced two-stage classifier” Computer Networks Volume 250, August 2024, 110576

[57] Othman, S.M., Ba-Alwi, F.M., Alsohybe, N.T. et al. Intrusion detection model using machine learning algorithm on Big Data environment. J Big Data 5, 34 (2018). https://doi.org/10.1186/s40537-018-0145-4

[58] Hameed, Bashar; AlHabshy, AbdAllah; and ElDahshan, Kamal (2021) "Distributed Intrusion Detection Systems in Big Data: A Survey," Al-Azhar Bulletin of Science: Vol. 32: Iss. 1, Article 14. DOI: https://doi.org/10.21608/absb.2021.63810.1100

[59] Lamon, Kurt, "Intrusion detection for industrial control systems" (2021). EWU Masters Thesis Collection. 662. https://dc.ewu.edu/theses/662

[60] Chenxi Xiong “Hybrid Feature Selection In Network Intrusion Detection Using Decision Tree” August 2020

[61] Gutierrez-Garcia, Jose & Sánchez-DelaCruz, Eddy & Pozos-Parra, Maria. (2023). A Review of Intrusion Detection Systems Using Machine Learning: Attacks, Algorithms and Challenges. 10.1007/978-3-031-28073-3_5.

[62] Vadhil, Fatimetou & Salihi, Mohamed & Nanne, Mohamedade. (2024). Machine learning-based intrusion detection system for detecting web attacks. IAES International Journal of Artificial Intelligence (IJ-AI). 13. 711. 10.11591/ijai.v13.i1.pp711-721.

[63] Dorothy, Beatrice. (2024). Sql Injection Attack Detection Using Machine Learning Methods. International Research Journal of Modernization in Engineering Technology and Science. 6. 2813-2816. 10.56726/IRJMETS48788.

[64] Gupta, Anish & Tyagi, Lalit & Mohamed, Aezeden. (2023). A Machine Learning Methodology for Detecting SQL Injection Attacks. 184-191. 10.1109/ICTACS59847.2023.10390153.

[65] Mustapha, Adeyinka & Oteniara, Aminat & Ashi, Thomas. (2024). Comprehensive review of machine learning models for SQL injection detection in e-commerce. 10.13140/RG.2.2.14636.27520.

[66] Mr. Anmol Adhikari et.al “Machine Learning Technique for Intrusion Detection in the field of the Intrusion Detection System” July 2023

[67] Abdulkareem, Ademola & Somefun, Tobiloba & Adesina, Lambe & Adewale, Adeyinka. (2024). Experimental analysis of intrusion detection systems using machine

learning algorithms and artificial neural networks. International Journal of Electrical and Computer Engineering (IJECE). 14. 983. 10.11591/ijece.v14i1.pp983-992.

[68] Aliyu, Garba & Thandekkattu, Salu & Abdulmumin, Idris & Baba, Usman & Yusuf, Aminat & Nasir, Mustapha. (2022). Machine Learning Based Intrusion Detection on Complex Nested Transactional SQL Queries. 10.1109/NIGERCON54645.2022.9803152.

[69] Janabi, Ahmed & Kanakis, Triantafyllos & Johnson, Mark. (2023). A Survey of Intrusion Detection Systems Based Machine Learning Approaches Applied to Software-Defined Networks (SDN): Research Issues and Challenges. 10.20944/preprints202312.1449.v1.

[70] Kumari, Nidhi. (2022). Intrusion Detection System Using Machine Learning: An Overview. 10.13140/RG.2.2.20109.41441.

[71] Pereira, Danillo & Silva, F. & MolinaSapia, Helton & Kakihata, Eduardo & Oikawa, Ronaldo. (2017). Intrusion Detection System Based On Flows Using Machine Learning Algorithms. IEEE Latin America Transactions. 15. 1998.

[72] Al, Mohammed & Alenezi, Freeh & Lu, Zhuo. (2019). Effectiveness of Machine Learning Based Intrusion Detection Systems. 10.1007/978-3-030-24907-6_21.

[73] Barhoush, Malek & Jaradat, Ameera & Easa, Rawan. (2022). Network Intrusion Detection System: Machine Learning Approach. Indonesian Journal of Electrical Engineering and Computer Science. xx. 1151-1158. 10.11591/ijeecs.v25.i2.pp1151-1158.

[74] Gandhar, Abhishek & Priyadarshi, Prakhar & Gandhar, Shashi & Kumar, Shashi & Rehalia, Arvind & Tiwari, Mohit. (2024). An Advanced design of Intrusion Detection System using Machine learning Architecture. Library Progress (International). Vol. 44. 15925-15935.

[75] Joshi, Anamika & Geetha, Vasantha. (2014). SQL Injection detection using machine learning. 2014 International Conference on Control, Instrumentation, Communication and Computational Technologies, ICCICCT 2014. 1111-1115. 10.1109/ICCICCT.2014.6993127.

[76] Kumar, Satish & Gupta, Sunanda & Arora, Sakshi. (2021). Research Trends in Network-Based Intrusion Detection Systems: A Review. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3129775.

[77] Saba, Tanzila & Sadad, Tariq & Rehman, Amjad & Mehmood, Zahid & Javaid, Qaiser. (2020). Intrusion Detection System (IDS) through Advance Machine Learning for the Internet of Things (IoT) Networks. IT Professional. 23. 10.1109/MITP.2020.2992710.

[78] Benmalek, Mourad & Haouam, Kamel-Dine. (2024). Advancing Network Intrusion Detection Systems with Machine Learning Techniques. Advances in Artificial Intelligence and Machine Learning. 4. 2575-2592. 10.54364/AAIML.2024.43150.

[79] Sultan, Muhammad & Hira, Sammia & Bhatti, Sohail & Rakha, Allah. (2024). Analysis of Machine Learning Methods for Intrusion Detection Systems in Wireless Networks. Bulletin of Business and Economics (BBE). 13. 391-403. 10.61506/01.00343.

[80] Kiran, K.V.V.N.L. & Devisetty, R.N. & Kalyan, N. & Mukundini, K. & Ramesh, Karthi. (2020). Building a Intrusion Detection System for IoT Environment using Machine Learning Techniques. Procedia Computer Science. 171. 2372-2379. 10.1016/j.procs.2020.04.257.

[81] Ahmed, Md & Islam, Salekul & Shatabda, Swakkhar & Islam, A.K.M. & Robin, Md Towhidul Islam. (2021). Intrusion Detection System in Software-Defined Networks Using Machine Learning and Deep Learning Techniques –A Comprehensive Survey. 10.36227/techrxiv.17153213.v1.

[82] Desai, Pranjali & Sonawane, Avantika & Mane, Tanvi & Jaiswal, Rupesh. (2023). NETWORK BASED INTRUSION DETECTION SYSTEM. 5. 3851-3857. 10.56726/IRJMETS35232.

[83] Auninda Alam et.al “SCAMM: Detection and Prevention of SQL Injection Attacks Using a Machine Learning Approach” January 2021

[84] Alrowaily, Mohmmed, "Investigation of Machine Learning Algorithms for Intrusion Detection System in Cybersecurity" (2020). USF Tampa Graduate Theses and Dissertations. https://digitalcommons.usf.e'du/etd/8915

[85] Sodiq Idowu Ibrahim et.al "Automatic Detection Of Injecton Attack In Http Requests" June, 2019

[86] Ahmad Maroof Karimi et.al "Distributed Machine Learning Based Intrusion Detection System" August 2016

[87] Mostofa Kamrul Ahsan et.al "Increasing The Predictive Potential Of Machine Learning Models For Enhancing Cybersecurity" November 2020

[88] Eric M Nyakundi "Using Support Vector Machines in Anomaly Intrusion Detection"2015

[89] Sharafaldin, I., Lashkari, A. and Ghorbani, A. Toward Generating a New Intrusion Detection Dataset and Intrusion Traffic Characterization. DOI: 10.5220/0006639801080116

[90] Ali, H.A., El-Mikkawy, M.E., & Barakat, S.I. (2011). Database Intrusion Detection System.

[91] Stephen Ngure Gitonga et.al "Machine Learning Model for Denial of Service Network Intrusion Detection" International Research Journal of Innovations in Engineering and Technology (IRJIET) ISSN (online): 2581-3048 Volume 3, Issue 7, pp 22-28, July-2019

[92] Yu Shen "Machine Learning and Knowledge-Based Integrated Intrusion Detection Schemes" 2022

[93] Sharma, Sushant & Zavarsky, Pavol & Butakov, Sergey. (2020). Machine Learning based Intrusion Detection System for Web-Based Attacks. 227-230. 10.1109/BigDataSecurity-HPSC-IDS49724.2020.00048.

[94] Fatani, Abdulaziz & Elsayed Abd Elaziz, Mohamed & Dahou, Abdelghani & Al-qaness, Mohammed & Lu, Songfeng. (2021). IoT Intrusion Detection System Using

Deep Learning and Enhanced Transient Search Optimization. IEEE Access. PP. 1-1. 10.1109/ACCESS.2021.3109081.

[95] S. Garc´ıa, M. Grill, J. Stiborek, and A. Zunino, “An empirical comparison of botnet detection methods,” Computers & Security, vol. 45, pp. 100–123, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167404814000923

[96] H. G. Kayacik, A. N. Zincir-Heywood, and M. I. Heywood, “Selecting features for intrusion detection: A feature relevance analysis on KDD 99,” in Third Annual Conference on Privacy, Security and Trust Proceedings, 2005. [Online]. Available: http://www.lib.unb.ca/Texts/PST/2005/pdf/kayacik.pdf

[97] S. Abt and H. Baier, “Are we missing labels? a study of the availability of ground-truth in network security research,” in 2014 Third International Workshop on Building Analysis Datasets and Gathering Experience Returns for Security (BADGERS). IEEE, 2014, pp. 40–55.

[98] J. McHugh, “Testing intrusion detection systems: A critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory,” ACM Trans. Inf. Syst. Secur., vol. 3, no. 4, p. 262–294, nov 2000. [Online]. Available: https://doi.org/10.1145/382912.382923

[99] L. Breiman, “Bagging Predictors,” Machine Learning, vol. 24, no. 2, pp. 123–140, 1996.

[100] L. Rokach, “Taxonomy for characterizing ensemble methods in classification tasks: A review and annotated bibliography,” Computational Statistics & Data Analysis, vol. 53, no. 12, pp. 4046–4072, 2009. [Online]. Available: https://www.sciencedirect.com/science/article/pii/ S0167947309002631

[101] K. Sindhya, K. Miettinen, and K. Deb, “A hybrid framework for evolutionary multi-objective optimization,” IEEE Transactions on Evolutionary Computation, vol. 17, no. 4, pp. 495–511, 2013.

[102] S. Zhang, H. Wang, D. Yang, and M. Huang, “Hybrid Multi-Objective Genetic Algorithm for Multi-Objective Optimization Problems,” in The 27th Chinese Control and Decision Conference (2015 CCDC). IEEE, 2015, pp. 1970–1974.

[103] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen, Evolutionary Algorithms for Solving Multi-Objective Problems. Springer New York, NY, 2007.

[104] M. Garuba, C. Liu, and D. Fraites, “Intrusion techniques: Comparative study of network intrusion detection systems,” Proceedings - International Conference on Information Technology: New Generations, ITNG 2008, pp. 592–598, 2008.

[105] L. Oz, E. Eilertson, A. Lazarevic, P.-n. Tan, V. Kumar, and J. Srivastava, “Minds - minnesota intrusion detection system,” pp. 1–21, 08 2003. [Online]. Available: https://www.researchgate.net/publication/ 2878372 MINDS - Minnesota Intrusion Detection System

[106] Z. Chunyue, L. Yun, and Z. Hongke, “A pattern matching based network intrusion detection system,” in 2006 9th International Conference on Control, Automation, Robotics and Vision. IEEE, 2006, pp. 1–4.

[107] Y. F. Hsu, Z. Y. He, Y. Tarutani, and M. Matsuoka, “Toward an online network intrusion detection system based on ensemble learning,” IEEE International Conference on Cloud Computing, CLOUD, vol. 2019-July, pp. 174–178, 2019.

[108] J. Yang, J. Deng, S. Li, and Y. Hao, “Improved traffic detection with support vector machine based on restricted Boltzmann machine,” Soft Computing, vol. 21, no. 11, pp. 3101–3112, 2017.

[109] M. A. Jabbar, R. Aluvalu, and S. S. Reddy, “RFAODE: A Novel Ensemble Intrusion Detection System,” Procedia Computer Science, vol. 115, pp. 226–234, 2017. [Online]. Available: https: //doi.org/10.1016/j.procs.2017.09.129

[110] B. A. Tama, L. Nkenyereye, S. R. Islam, and K.-S. Kwak, “An enhanced anomaly detection in web traffic using a stack of classifier ensemble,” IEEE Access, vol. 8, pp. 24 120–24 134, 2020.

[111] H. Rajadurai and U. D. Gandhi, “A stacked ensemble learning model for intrusion detection in wireless network,” Neural Computing and Applications, vol. 34, no. 18, p. 15387–15395, 2020. [Online]. Available: https://doi.org/10.1007/s00521-020-04986-5

[112] M. M. Lisehroodi, Z. Muda, W. Yassin, and N. Izura Udzi, “KMNEU: An Efficient Hybrid Approach for Intrusion Detection System,” Research Journal of Information Technology, vol. 6, no. 1, pp. 46–57, jan 2014. [Online]. Available: http://www.scialert.net/abstract/?doi= rjit.2014.46.57

[113] Waikato University, “Machine Learning Project at the University of Waikato in New Zealand,” 2021. [Online]. Available: https: //www.cs.waikato.ac.nz/ml/index.html

[114] R. C. Holte, “Very simple classification rules perform well on most commonly used datasets,” Mach. Learn., vol. 11, no. 1, p. 63–90, apr 1993. [Online]. Available: https://doi.org/10.1023/A:1022631118932

[115] X. XU, “ConjunctiveRule.” [Online]. Available: https:// weka.sourceforge.io/doc.packages/conjunctiveRule/

[116] C. Brown and A. Cowperthwaite, “Analysis of the 1999 darpa/lincoln laboratory ids evaluation data with netadhict.” IEEE, jul 2009, pp. 1–7. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber= 5356522http://ieeexplore.ieee.org/xpls/abs{ }all.jsp?arnumber=5356522

[117] R. Lippmann, “Proposed 1999 DARPA Off-line Intrusion Detection Evaluation Plans Overview of Last Year ’ s 1998 Evaluation,” pp. 1–11, 1999.

[118] I. Information and Computer Science, University of California, “KDD Cup 1999 Data,” 1999. [Online]. Available: http://kdd.ics.uci.edu/databases/ kddcup99/kddcup99.html

[119] 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, July 2009, pp. 1–6.

[120] Proebstel, E. P. (2008). Characterizing and improving distributed network-based intrusion detection systems(nids):timestamp synchronization and sampled traffic. Master's thesis, University of California DAVIS, CA, USA.

[121] R. Chitrakar, C. H. (2012). Anomaly based intrusion detection using hybrid learning approach of combining kmedoids clustering and naive bayes classification. In 8th WiCOM, pages 1–5.

[122] Sangster, B., OConnor, T. J., Cook, T., Fanelli, R., Dean, E., Adams, W. J., Morrell, C., and Conti, G. (2009). Toward instrumenting network warfare competitions to generate labeled datasets. In 2009 Usenix. Usenix: The Advanced Computing System Association.

[123] Scott, P. and Wilkins, E. (1999). Evaluating data mining procedures: techniques for generating artificial data sets. Information and Software Technology, 41(9):579 – 587.

[124] Sharafaldin, I., Gharib, A., Habibi Lashkari, A., , and Ghorbani, A. A. (2017). Towards a reliable intrusion detection benchmark dataset. Software Networking, 2017:177–200.

[125] Song, J., Takakura, H., Okabe, Y., Eto, M., Inoue, D., and Nakao, K. (2011). Statistical analysis of honeypot data and building of kyoto 2006+ dataset for nids evaluation. In Proceedings of the First Workshop on Build

[126] Heidemann, J. and Papdopoulos, C. (2009). Uses and challenges for network datasets. In Cybersecurity Applications Technology Conference For Homeland Security,CATCH'09, pages 73–82.

[127] Koch, R., Golling, M. G., and Rodosek, G. D. (2017). Towards comparability of intrusion detection systems: New data sets. In Proceedings of the TERENA Networking Conference, page 7.

[128] M. Sato, H. Yamaki, H. T. (2012). Unknown attacks detection using feature extraction from anomaly-based ids alerts. In Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on, pages 273–277.

[129] McHugh, J. (2000). Testing intrusion detection systems: A critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. ACM Trans. Inf. Syst. Secur., 3(4):262–294.

[130] Nechaev, B., Allman, M., Paxson, V., and Gurtov, A. (2004). Lawrence berkeley national laboratory (lbnl)/icsi enterprise tracing project.

[131] Nehinbe, J. O. (2010). A Simple Method for Improving Intrusion Detections in Corporate Networks, pages 111–122. Springer Berlin Heidelberg.

[132] Nehinbe, J. O. (2011). A critical evaluation of datasets for investigating idss and ipss researches. In IEEE 10th International Conference on CIS, pages 92–97.

[133] Ali Shiravi, Hadi Shiravi, M. T. and Ghorbani, A. A. (2012). Toward developing a systematic approach to generate benchmark datasets for intrusion detection. Computers and Security, 31(3):357 – 374.

[134] Brown, C., Cowperthwaite, A., Hijazi, A., and Somayaji, A. (2009). Analysis of the 1999 darpa/lincoln laboratory ids evaluation data with netadhict. In 2009 IEEE SCISDA, pages 1–7.

[135] CICFlowMeter (2017). Canadian institute for cybersecurity (cic).

[136] Creech, G. and Hu, J. (2013). Generation of a new ids test dataset: Time to retire the kdd collection. In 2013 IEEE Wireless Communications and Networking Conference (WCNC), pages 4487–4492.

[137] V. Jyothsna, V. V Rama Prasad, and K. Munivara Prasad, “A Review of Anomaly based Intrusion Detection Systems,” Int. J. Comput. Appl., vol. 28, no. 7, pp. 26–35, 2011.

[138] B. Schölkopf, “An Introduction to Support Vector Machines,” in Recent Advances and Trends in Nonparametric Statistics, 2003.

[139] Bhargava and Sharma, “Decision Tree Analysis on J48 Algorithm for Data Mining,” Int. J. Adv. Res. Decis. Tree Anal. J48 Algorithm Data Min., 2013.

[140] G. P. Rout and S. N. Mohanty, “A hybrid approach for network intrusion detection,” in Proceedings - 2015 5th International Conference on Communication Systems and Network Technologies, CSNT 2015, 2015

[141] C. F. Tsai, Y. F. Hsu, C. Y. Lin, and W. Y. Lin, “Intrusion detection by machine learning: A review,” Expert Systems with Applications. 2009

[142] E. Darra and S. K. Katsikas, “A survey of intrusion detection systems in wireless sensor networks,” in Intrusion Detection and Prevention for Mobile Ecosystems, 2017.

[143] Nunez, J.C., Cabido, R., Pantrigo, J.J., Montemayor, A.S. and Velez, J.F., 2018. Convolutional neural networks and long short-term memory for skeleton-based human activity and hand gesture recognition. Pattern Recognition, 76, pp.80-94. 59.

[144] Powers, D.M., 2020. Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. arXiv preprint arXiv:2010.16061. 60.

[145] Lakhina, S., Joseph, S. and Verma, B., 2010. Feature reduction using principal component analysis for effective anomaly–based intrusion detection on NSL-KDD. 61.

[146] Benaddi, H., Ibrahimi, K. and Benslimane, A., 2018, October. Improving the intrusion detection system for nsl-kdd dataset based on pca-fuzzy clustering-knn. In 2018 6th International Conference on Wireless Networks and Mobile Communications (WINCOM) (pp. 1-6). IEEE. 62.

[147] Ikram, S.T. and Cherukuri, A.K., 2016. Improving accuracy of intrusion detection model using PCA and optimized SVM. Journal of computing and information technology, 24(2), pp.133-148. 63.

[148] Gao, J., Chai, S., Zhang, B. and Xia, Y., 2019. Research on network intrusion detection based on incremental extreme learning machine and adaptive principal component analysis. Energies, 12(7), p.1223. 64.

[149] Kumar, V., Sinha, D., Das, A.K., Pandey, S.C. and Goswami, R.T., 2020. An integrated rule based intrusion detection system: analysis on UNSW-NB15 data set and the real time online dataset. Cluster Computing, 23(2), pp.1397-1418.

[150] Vanin, Patrick & Newe, Thomas & Dhirani, Lubna Luxmi & O'Connell, Eoin & O'Shea, Donna & Lee, Brian & Rao, Muzaffar. (2022). A Study of Network Intrusion Detection Systems Using Artificial Intelligence/Machine Learning. Applied Sciences. 12. 11752. 10.3390/app122211752.

Controller General of Patents, Designs & Trade Marks
CP-2, Sector V, Salt Lake City, Kolkata-700091
Tel No. (091)(033) 23671945-46 Fax No. 033 23671988
E-mail: kolkata-patent@nic.in
Web Site: www.ipindia.gov.in



सत्यमेव जयते

INTELLECTUAL PROPERTY INDIA
PATENTS | DESIGNS | TRADE MARKS
GEOGRAPHICAL INDICATIONS

G.A.R.6
[See Rule 22(1)]
RECEIPT

Date/Time 11/04/2026

To
Vikramsingh Ravindrasingh Parihar

61, Mahesh Nagar, near Sainagar, Amravati

User Code: vikrampari

User Name: Vikramsingh Ravindrasingh Parihar

CBR Detail:

Sr. No.	Ref. No./Application No.	App. Number	Amount Paid	C.B.R. No.	Form Name	Remarks
1	498177-001		1000	208793	FORM 1	Graphical User Interface for SQL Database Intrusion Detection System

TransactionID	Payment Mode	Challan Identification Number	Amount Paid	Head of A/C No
D-0000143472	Online Bank Transfer	1104260011948	1000.00	1475001020000001

Total Amount : ₹ 1000
Amount in Words: Rupees One Thousand Only

Received from Vikramsingh Ravindrasingh Parihar the sum of ₹ 1000 on account of Payment of fee for above mentioned Application/Forms.

* This is a computer generated receipt, hecnce no signature required.

Print

Home

भारत सरकार / GOVERNMENT OF INDIA
कॉपीराइट कार्यालय / Copyright Office
बौद्धिक संपदा भवन,प्लॉट संख्या 32, सेक्टर 14, द्वारका, नई दिल्ली-110078 फोन: 011-28032496
Intellectual Property Bhawan, Plot No. 32, Sector 14, Dwarka, New Delhi-110078 Phone: 011-28032496
रसीद / Receipt

Copyright Office

To,

Amarnath-Chadchankar
Narhe--411041

(M)-9975474722 : E-mail- amarnathchadchankar@gmail.com

RECEIPT NO : 245683
FILING DATE : 28/05/2026
BRANCH : Delhi
USER : Amarnath38

S.No	Form	Diary No.	Request No	Title	Amount (Rupees)
1	Form-XIV	LD-25177/2026-CO	279063	Correlated Preprocessing and Analysing for Intrusion Detection System for SQL Databases using Machine Learning Technique.	500
Amount in Words			Rupees Five Hundreds		500

PAYMENT MODE	Transaction Id	CIN
Online	C-0000319617	2805260075742

(Administrative Officer)

*This is a computer genereated receipt, hence no signature required.

*Please provide your email id with every form or document submitted to the Copyright office so that you may also receive acknowledgements and other documents by email.

ISSN: 2584-282X

INTERNATIONAL JOURNAL OF TRENDS IN EMERGING RESEARCH AND DEVELOPMENT

Publication Certificate

This is to certify that

Authored by

Chadchankar Amarnath Shivan

In the recognition of the paper titled **"Intrusion detection systems for SQL databases using machine learning"** has been published in

Volume **2** Issue **6** Page Number: **257-261**, **International Journal of Trends in Emerging Research and Development.**

Certificate Number: 3-5-16

Authorized By:
Publisher: **Vipin**

International Journal of Trends in Emerging Research and Development
Vipin

Publisher: Vipin
Address: Z-372, Prem Nager, Kirari Suleman Nagar, North West Delhi, Delhi-110086, Delhi-110086, India
Contact No.: +91 9315510518
Email.: editor.ijterdjournal@gmail.com
Website.: https://researchtrendsjournal.com/

Online at: https://researchtrendsjournal.com
Indexed Journal

ISSN No: 2584-282X
Peer Reviewed Journal



INTERNATIONAL JOURNAL OF TRENDS IN EMERGING RESEARCH AND DEVELOPMENT

Volume 2; Issue 6; 2024; Page No. 257-261

Received: 08-08-2024
Accepted: 19-10-2024

Intrusion detection systems for SQL databases using machine learning

[1]Chadchankar Amarnath Shivanand, [2]Dr. Balveer Singh, [3]Dr. Yashpal Singh, [4]Dr. Rohita Yamaganti and [5]Dr. Swati Dey

[1]Research Scholar, P.K. University, Shivpuri, Madhya Pradesh, India
[2-5]Professor, P.K. University, Shivpuri, Madhya Pradesh, India

DOI: https://doi.org/10.5281/zenodo.17132174

Corresponding Author: Chadchankar Amarnath Shivanand

Abstract

SQL Injection is still among the worst security flaws, exposing users' private data and causing financial losses. The most current OWASP Top 10 study ranks injection attacks as the top vulnerability, and the frequency of these attacks is on the rise. By definition, Intrusion Detection System (IDS) rules that rely on static signatures are a common component of traditional defensive systems. These rules are great for preventing known attacks but aren't effective against zero-day threats. Many recent studies have used machine learning methods, which may identify previously unseen threats but can be performance-heavy depending on the methodology. To add insult to injury, some intrusion detection systems scour database server logs for information, while others focus on gathering traffic entering the web app across the network or the web app host. A web application host, a MySQL database server, and a Datiphy appliance node placed between the two are the two sources of traffic that will be collected in this project. Through our examination of these two datasets as well as an additional dataset that is correlated with them, we have proven that the accuracy achieved with the correlated dataset using algorithms like decision trees and rule-based approaches is comparable to that of a neural network algorithm, but with substantially better performance.

Keywords: Industry, traditional, machine learning, performance, companies

Introduction

Web assaults like SQL Injection have been around for a long time, but they still endanger individuals' private information as well as the security of companies and governments suffer financial losses as a result. That is especially the case when considering new attack vectors are always appearing and old ones are constantly evolving. Web attack mitigation receives substantial funding from industry and security businesses; yet, many existing mitigation solutions have limits that are being actively sought to be overcome by current research.

Static analysis of incoming online traffic, often called one typical approach to traditional online attack mitigation is signature detection. Creating a signature that is specific to cyberattacks is the essence of this strategy; when a firewall or other security appliance detects this signature, potentially obstruct the suspicious traffic. Although this method is quick and may be used in real-time to protect network assets, it does have one drawback: it can only identify known assaults.

One other way to protect against SQL injection attacks on the web is to examine the syntax of incoming SQL queries; an attack of this kind is identified when a query is found to be erroneous. Unfortunately, this technique requires extensive understanding of the program and the framework of "ordinary" questions; yet, it manages to do a fantastic job of detecting novel threats using incorrect queries and has good detection results overall.

One approach to detecting SQL injections that is now being investigation use techniques derived from machine learning. The employment of SVMs, decision trees, neural networks, and rule-based learning techniques is common in this field of study. The ability to identify novel assaults is a major strength of these methods. One possible downside of these strategies is that, depending on the algorithm utilized, processing time might potentially rise.

Intrusion Detection Systems

One of the main functions of an IDS is to keep an eye out for any suspicious or illegal behavior on a network or inside its systems. Usually, In the event that a security incident occurs, a SIEM system will either alert the administrator or compile all of the pertinent data into a single location of an incursion or breach. By combining data from many sources and using alert filtering algorithms, a SIEM system can distinguish between legal and hazardous activity.

Intrusion detection systems' many spheres of influence vary from individual PCs to extensive networks. These days, NIDSs and HIDSs are the two main categories used to describe these kinds of systems. An HIDS would be a system that keeps tabs on critical OS files, whereas an NIDS would be one that examines all incoming network traffic. Another way to categorize IDS is by detection method. Two of the most popular variations are signature-based detection and anomaly-based detection are two methods for detecting harmful patterns and abnormal traffic, respectively. The former employs machine learning to find signatures, while the latter analyses historical data to identify deviations from a model of "good" traffic. One more popular variation is reputation-based detection, which involves identifying possible threats based on their reputation ratings. The capacity to react to detected intrusions is a feature of several IDS solutions. Intrusion prevention systems (IPS) are often used to describe systems that can respond to potential intrusions. The use of specialized tools may also make intrusion detection systems more useful; for example, a honeypot can be used to detect and identify malicious traffic.

Machine Learning

Machine learning tasks

There are primarily two stages to any creating an algorithm for machine learning. Different models have different ways of being trained. Common frameworks used for this purpose and for designing machine learning models include scikit-learn, Tensor flow, PyTorch, Matlab, and Weka. When it comes to optimizing the method or the number of parameters accessible, the framework is king.

Among the various applications of machine learning models, intrusion detection finds special attention in three: classification, regression, and reconstruction. Entries are sorted into many classifications by classification, which might include "normal" or "attack" or even distinct families of assaults. In order to find continuous values, such is the possibility that a given input constitutes an assault, regression (also referred to as "prediction") is used. Lastly, only a certain class of neural networks can do reconstruction. In order to get representation learning allows the network to acquire the characteristics, this job attempts to decompress and recompress the input data in an effort to rebuild it.

Machine learning algorithms may be trained in one of two ways: supervised or unsupervised. Neural networks are one example of a model that may be taught in either direction. The majority of models are trained via supervised learning, where the dataset includes inputs and the respective correct results. Assigning these outputs to their respective inputs is a mathematical function, which the algorithm learns to represent. Classification and regression are two classic supervised training problems. Contrarily, findings from the training dataset are not used in unsupervised training. Familiarizing oneself with intriguing data structures is its goal. An example of an activity that does not need supervision is reconstruction.

Machine learning models need testing once training is complete in order to evaluate their efficacy. Data that wasn't in the training set can't be used for this assessment. If the model had seen this data before—or the right outcomes in the case of supervised learning—then the assessment would be skewed. You can also compare alternative parameters' values (for example, a learning rate or neurone count) using a validation set. Finding the value that performed the subsequent top performer on the validation set step after training. We next put the whole network through its paces on the test data. Additionally, fresh data is required for a validation set; typically, a subset set aside specifically for this purpose.

Table 1: Supervised learning vs unsupervised learning

	Supervised learning	**Unsupervised learning**
Process	Input and output data are given.	Only input data is given.
Input data	The machine is trained using labeled data.	The machine is not given unlabeled data.
Algorithms used	SVM, NN, Random Forest, Linear and Logistics regression, Classification trees	Different categoriez: K-means, Cluster algorithms, Hierarchical clustering, and so on.
Computational complexity	Simple.	Complex.
USA of data	Uses training data and relate input and output results.	Does not use output data.
Accuracy of results	Accurate and trustworthy.	Less accurate and trustworthy.
Real time learning	Learning is offline.	Real-time.
Number of classes	Known.	Unknown.
Main drawbacks	Big data is a challenge.	No precise information in regards to data sorting, and the output is not known.

Literature Review

Deepa Manikandan *et al*. (2024) [1] Security across all domains, including databases, networks, and the cloud, has grown in importance in real-time distributed systems since the advent of cyberspace. When faced with evolving threats, present-day intrusion detection systems (IDS) find it difficult to stay involved. The proposed model DR-DBMS (dimensionality reduction in database management systems) combines supervised machine learning methods, dimensionality reduction approaches, and sophisticated rule-based classifiers to enhance the accuracy of intrusion detection for different kinds of assaults. Simulation findings show that the DR-DBMS system effectively used methods for dimensionality reduction and traits selection for

determining intrusion assault in 0.07 seconds with a lesser number of characteristics.

Roland Plaka (2021) [2] Intrusion detection systems (IDS) are designed to keep an eye out for any breaches in a network's confidentiality, integrity, or availability that may have occurred as a result of malicious or unapproved activity. Machine learning methods for detection and classification, a variety of intrusion detection systems (IDS), and anomaly detection approaches were thoroughly reviewed in this thesis. We provide an architecture for intrusion detection systems (IDS) that combines traditional methods with machine learning techniques. The current detection methods may be enhanced for better attack detection and categorization by including suitable machine learning techniques. Additionally, we have made an effort to evaluate and execute a battery of virtual tests on each machine learning algorithm to compare their performance. For general intrusion detection systems used in industrial control applications, our method offers indications for choosing machine learning techniques.

Yasmeen S. Almutairi *et al*. (2022) [3] Safeguarding sophisticated communication networks requires intrusion detection systems (IDS). Certain patterns, signatures, and rule breaches were the primary targets of these systems' design. Potential new methods to network intrusion detection have emerged Using Deep Learning and Machine Learning methods in the last many years. Techniques like this may tell the difference between regular patterns and those that aren't. Support Vector Machine, J48, Random Forest, and Naïve Bytes were among the machine learning techniques used to assess the Network Intrusion Detection Systems (NIDS) in this article. The methods used binary and multi-class categorization on the NSL-KDD benchmark data set. Extensive discussion is provided about the outcomes of using such strategies, which exceeded our earlier efforts.

Amit Singh *et al*. (2024) [4] Intrusion detection systems (IDS) are facing new hurdles due to encrypted data, new protocol variety, and an increase in the amount of criminal actions globally. Intrusion detection systems that rely on signatures are now at a stage where inadequate in this case. Several academics have put forth IDSs that use machine learning to detect intrusions previously unseen harmful actions by analysing patterns of activity. Intrusion detection systems that use on machine learning have many advantages than SIDS that rely on signatures are more effective in spotting novel forms of network-based malware. This study evaluated network intrusion detection systems utilising the IDS dataset, which contains the most recent common attacks, and two data resampling techniques in addition to 10 machine learning classifiers. The top three IDS models-XGBoost, KNneighbors, and AdaBoost-perform better than binary-class classification with 99.49%, 99.14%, and 98.75% accuracy, respectively. XGBoost, KNneighbors, and GaussianNB are the most accurate in multi-class classification with 99.30%, 98.88%, and 96.66% accuracy, respectively.

Tahsinur Rahman (2022) [5] The need of security measures to prevent breaches is growing in direct correlation with the proliferation of internet-connected devices. The purpose of an Intrusion Detection System (IDS) is to identify and block potentially harmful network traffic. The need for anomaly-based intrusion detection systems that use machine learning to identify more recent assaults has arisen because new threats may easily bypass standard signature-based IDS. This thesis will focus on a specific theme: anomaly-based intrusion detection systems that use deep learning. A comparison is made between adversarial machine learning methods and Generative Adversarial Networks (GAN), with conventional deep learning techniques. We use statistical metrics on two separate datasets to assess the strategies. As part of the assessment process, malicious samples are taken into account alongside benign and known attack samples. The last step is to compare the optimal method to other open-source anomaly-based intrusion detection systems. Outperforming all other techniques was a method that used a GAN to generate adversarial samples. Furthermore, when faced with malicious data points, the method matches the performance of current anomaly-based IDS. After reviewing the literature, we have come to the conclusion that intrusion detection systems based on GANs may be enhanced to better withstand both new and malicious assaults.

Research Methodology

To determine whether incoming communication is benign or malicious, our proposed method in this study employs machine learning methods. A bespoke business chat web app running on a distant MySQL server forms the backbone of the system. There are two locations where data is recorded: first, in the HTTP traffic that flows between the servers that generate traffic and those that host the web applications. Secondly, in the data transfer between the webapp server and the offline database server using MySQL.

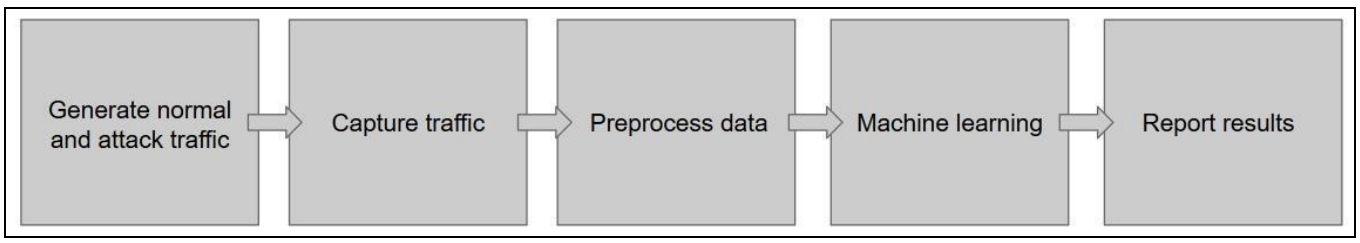


Fig 1: System Process

SCADA Dataset

After introducing the issue and ML methods utilized this section of the chapter aims to provide a general outline of the first dataset. This data collection originates from a CI water system that is controlled by SCADA.

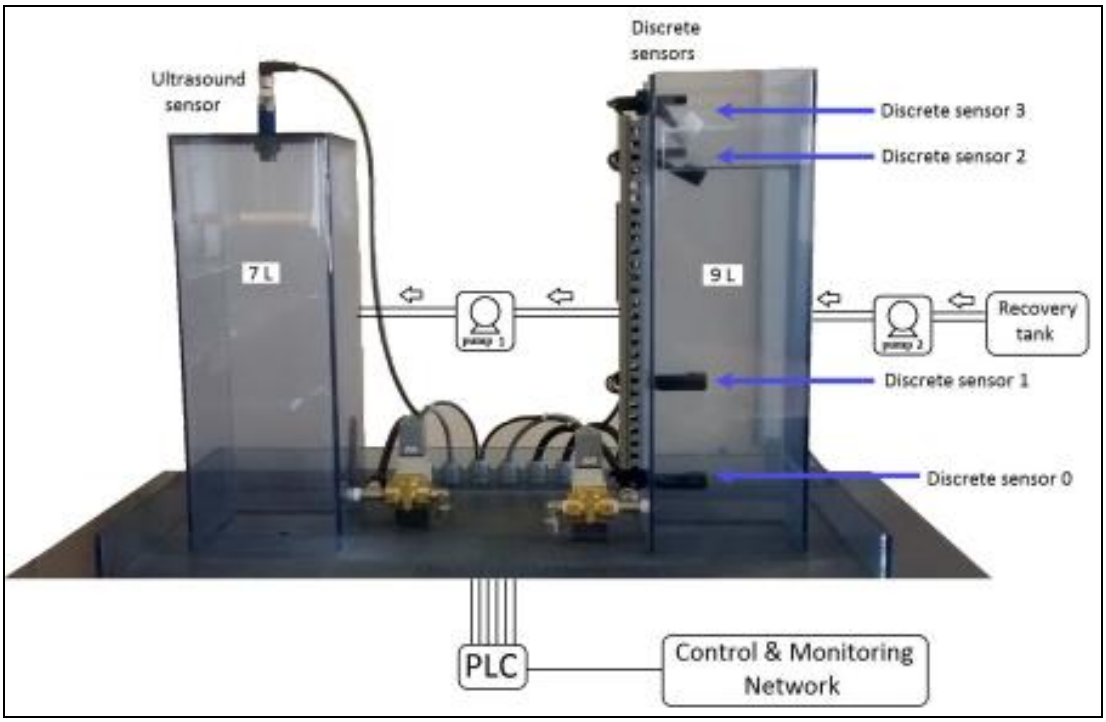


Fig 2: SCADA System Architecture

Experiment and Results

Table 1 summarizes the results of our research with several machine learning techniques. The time it takes to construct the machine learning models is called Model Time, the time it takes to classify the testing dataset using 5-fold cross-validation is called Testing Time, and Accuracy is the classification accuracy that each method achieves. Figure 1 displays the classification accuracy, whereas Table 2 displays the F-scores.

Table 2: Results With 20000 Records

Dataset	Algorithm	Accuracy	Model Time	Testing Time
Webapp	JRip	94.740%	3m30.85s	2.70s
	J48	95.630%	1m42.65s	2.55s
	RF	96.525%	5m30.20s	30.60s
	SVM	94.025%	2m35.80s	1m10.45s
	ANN	96.715%	46m03.55s	3.35s
Datiphy	JRip	95.980%	6m10.90s	3.35s
	J48	96.995%	1m59.30s	2.45s
	RF	97.210%	6m10.20s	33.50s
	SVM	95.190%	2m11.50s	1m3.45s
	ANN	97.285%	41m23.00s	2.95s
Correlated	JRip	97.150%	11m50.80s	2.10s
	J48	97.295%	2m01.80s	2.05s
	RF	98.055%	4m22.55s	35.45s
	SVM	95.715%	3m30.85s	1m5.90s
	ANN	97.615%	47m25.25s	3.80s

Scada

To demonstrate the precision of anomaly detection, this section details and assesses three separate studies. The purpose of the various tests is to offer varying degrees of detail about the presence of an abnormality. From just noting that an abnormality has occurred to pinpointing the offending part and the unusual circumstance, there is a wide range of possible outcomes.

Table 3: SCADA Results: Experiment 1 - Anomaly Detection (5-fold cross-validation)

Classification (Is Anomaly)	Recall	Precision	F1-Scpre
LR			
Benign	7.15%	90.34%	13.22%
Anomaly	99.89%	87.7%	93.4%
Weighted Average	87.73%	88.05%	88.05%
NB			
Benign	99.95%	16.74%	28.67%
Anomaly	24.99%	99.97%	39.98%
Weighted Average	34.82%	89.06%	89.06%
k-NN			
Benign	74.01%	79.7%	76.74%
Anomaly	97.15%	96.12%	96.63%
Weighted Average	94.12%	93.97%	93.97%
SVM			
Benign	7.15%	92.24%	13.23%
Anomaly	99.91%	87.70%	93.41%
Weighted Average	87.75%	88.30%	88.30%
Kernel SVM			
Benign	39.53%	98.52%	56.40%
Anomaly	99.91%	91.63%	95.59%
Weighted Average	91.99%	92.545£	92.54%
DT			
Benign	74.01%	74.72%	74.35%
Anomaly	96.22%	96.09%	96.15%
Weighted Average	93.30%	93.28%	93.28%
RF			
Benign	75.66%	75.99%	75.78%
Anomaly	96.38%	96.33%	96.36%
Weighted Average	93.67%	93.67%	93.67%

MQTT: We use the six ML algorithms-LR, Gaussian NB, k-NN, SVM, DT, and RF-discussed previously to evaluate various ML approaches on the MQTT-IoT-IDS2020 dataset. The following characteristics are disabled to prevent any identifying data from being impacted: protocol, MQTT flags, source and destination IP addresses.

Table 4: MQTT-IoT-IDS2020: Overall Detection Accuracy

	Packet	Features Unidirectional	Bidirectional
LR	78.87%	98.23%	99.44%
k-NN	69.13%	99.68%	99.9%
DT	88.55%	99.96%	99.95%
RF SVM (RBF Kernel)	65.39%	99.98%	99.97%
	77A%	97.96%	96.61%
NB SVM (Linear Kernel)	81.15%	78%	9755%
	66.69%	82.6%	98.5%

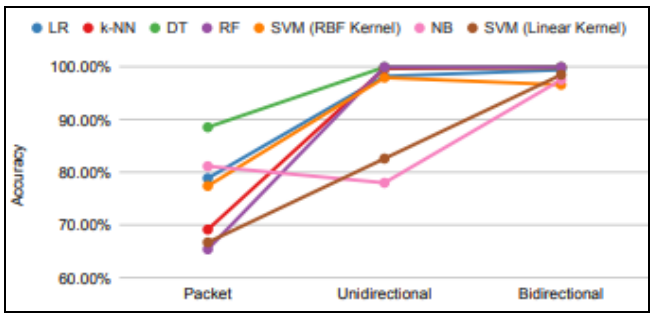


Fig 3: MQTT-IoT-IDS2020: Overall Detection Accuracy Trend using Different ML Techniques

Conclusion

Security of sensitive information, including financial and health records, remains a top concern due to SQL injection attacks and other web-based assaults. This problem is becoming more pressing as more and more social activities rely on the internet. We have demonstrated that the algorithms we have tested, including rule-based and decision tree algorithms, can classify testing data significantly faster and with accuracy comparable to that of Neural Networks, and we have also suggested a multi-source data analysis system to improve the accuracy of SQL injection attack detection. Adapting this system to detect other types of web-based attacks is on the list of future works. Other things on the list include gathering more data, like traffic going outbound from the web app to the browser, collecting larger datasets to see if that helps performance, and analyzing additional machine learning techniques for accuracy and performance.

Intrusion Detection Systems (IDS) are programs that scan every incoming and outgoing data packets for signs of malicious activity. In the last ten years, IDS have been built using a variety of ML approaches. The rapid emergence of new cyber threats has led to the widespread usage of ML approaches. This thesis delves into the use of ML approaches to construct IDS with specific purposes. In addition, this thesis explores the possibility of enhancing the efficiency and efficacy of next-generation IDS by using innovative DL approaches that have been successfully used in other areas of study.

Reference

1. Manikandan D, *et al*. Machine learning approach for intrusion detection system using dimensionality reduction. International Journal of Information Technology. 2022;34(1):1-7.
2. Plaka R, *et al*. Intrusion detection using machine learning for industrial control systems. Journal of Industrial Information Integration. 2021;24:100223.
3. Almutairi Y, Alhazmi B, Munshi A. Network intrusion detection using machine learning techniques. Advances in Science and Technology Research Journal. 2022;16(3):193-206. doi:10.12913/22998624/149934.
4. Singh A, Prakash J, Kumar G, Jain P, Ambati L. Intrusion detection system. Journal of Database Management. 2024;35(2). doi:10.4018/JDM.338276.
5. Rahman T. Intrusion detection system based on deep learning. 2022 Jul 29. [Thesis/Report].
6. Sama L. Network intrusion detection using deep learning. [Master's thesis]. Melbourne: Victoria University; c2022.
7. Kültür E. Network intrusion detection with a deep learning approach. 2022 Feb. 99 p. [Thesis/Report].
8. Worku G/Michael K. A predictive model of network intrusion detection systems using machine learning approach. 2023 Jan. [Report].
9. Raka M, et al. Intrusion detection using machine learning and log analysis. International Journal of Advanced Research in Computer and Communication Engineering. 2019;8(4):220-225.
10. Vanin P, Newe T, Dhirani LL, O'Connell E, O'Shea D, Lee B, Rao M. A study of network intrusion detection systems using artificial intelligence/machine learning. Applied Sciences. 2022;12(22):11752. doi:10.3390/app122211752.

Creative Commons (CC) License
This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY 4.0) license. This license permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

ISSN: 2583-9667

INTERNATIONAL JOURNAL OF ADVANCE RESEARCH IN MULTIDISCIPLINARY

Publication Certificate

This is to certify that

Authored by

Chadchankar Amarnath Shivan

In the recognition of the paper titled **"Advanced intrusion detection using ml on webapp and SQL traffic: A correlated dataset model"** has been published in Volume **3** Issue **3** Page Number: **267-271**, **Journal International Journal of Advance Research in Multidisciplinary.**

Certificate Number: 3-3-92

Authorized By:

Publisher: **Mrs. Nidhi**

Publisher: Mrs Nidhi
Address: Address: B-2/86A, Prem Nager, Kirari Suleman Nagar, North West Delhi, Delhi-110086, India
Contact No.: +9315510518
Email.: editor.ijarmjournals@gmail.com
Website.: https://multiresearchjournal.theviews.in

E-ISSN: 2583-9667
Indexed Journal
Peer Reviewed Journal
https://multiresearchjournal.theviews.in

Received: 14-04-2025
Accepted: 29-06-2025
Published: 16-07-2025

INTERNATIONAL JOURNAL OF ADVANCE RESEARCH IN MULTIDISCIPLINARY

Volume 3; Issue 3; 2025; Page No. 267-271

Advanced intrusion detection using ml on webapp and SQL traffic: A correlated dataset model

[1]Chadchankar Amarnath Shivanand, [2]Dr. Balveer Singh, [3]Dr. Yashpal Singh, [4]Dr. Rohita Yamaganti and [5]Dr. Swati Dey

[1]Research Scholar, P.K. University, Shivpuri, Madhya Pradesh, India
[2-5]Professor, P.K. University, Shivpuri, Madhya Pradesh, India

DOI: https://doi.org/10.5281/zenodo.17130376

Corresponding Author: Chadchankar Amarnath Shivanand

Abstract

A web application host, a MySQL database server, and a Datiphy appliance node placed between the two are the two sources of traffic that will be collected in this project. Through our examination of these two datasets as well as an additional dataset that is correlated with them, we have proven that the accuracy achieved with the correlated dataset using algorithms like decision trees and rule-based approaches is comparable to that of a neural network algorithm, but with substantially better performance. Newly interwoven technology and diverse systems have emerged as a result of the digital transformation of both general- and special-purpose networks., as our dependence on networked technologies has grown. Cyberattacks have grown in both frequency and sophistication in recent years, in part because of the dynamic given the current state of technology and the exponential growth of connected devices. The capacity to analyses network traffic using Intrusion Detection Systems (IDS) is a crucial component of any networking security toolset. Using ML and DL techniques, researchers have created intrusion detection systems (IDS) to cope with new and zero-day cyberattacks, which are becoming more common and sophisticated. But developing IDS is difficult since there aren't enough big, realistic, and current datasets.

Keywords: Techniques, machine learning, Intrusion Detection, reputation, cyberattacks

Introduction

One approach to detecting SQL injections that is now being investigation use techniques derived from machine learning. The employment of SVMs, decision trees, neural networks, and rule-based learning techniques is common in this field of study. The ability to identify novel assaults is a major strength of these methods. One possible downside of these strategies is that, depending on the algorithm utilized, processing time might potentially rise. Having access to reliable data is essential for studies investigating the aforementioned SQL detection strategies and others. Many recent studies have used online application and server logs, or web traffic that is collected as it enters the application. Our proposed approach makes use of information retrieved from a Datiphy appliance's database server and data retrieved from the web application's incoming and outgoing traffic on a network node. Using traffic data gathered at these two sites, we construct our first two datasets; by merging the events from the first two datasets, we get our third dataset. Improving the efficiency and speed of a classification system is difficult in any situation that calls for streaming enormous volumes of data, but especially in the intrusion detection issue area. In this research, we're investigating algorithms that are both quicker and more accurate than others, such as neural networks, but which may sacrifice efficiency in the process. To that end, we are always exploring new methods for improving signature detection's speed, efficiency, accuracy, and novelty of machine learning.

One of the main functions of an IDS is to keep an eye out for any suspicious or illegal behavior on a network or inside its systems. Usually, In the event that a security incident occurs, a SIEM system will either alert the administrator or compile all of the pertinent data into a single location of an incursion or breach. By combining data from many sources and using alert filtering algorithms, a SIEM system can distinguish between legal and hazardous activity. Intrusion detection systems' many spheres of influence vary from

individual PCs to extensive networks. These days, NIDSs and HIDSs are the two main categories used to describe these kinds of systems. An HIDS would be a system that keeps tabs on critical OS files, whereas an NIDS would be one that examines all incoming network traffic. Another way to categorize IDS is by detection method. Two of the most popular variations are signature-based detection and anomaly-based detection are two methods for detecting harmful patterns and abnormal traffic, respectively. The former employs machine learning to find signatures, while the latter analyses historical data to identify deviations from a model of "good" traffic. One more popular variation is reputation-based detection, which involves identifying possible threats based on their reputation ratings. The capacity to react to detected intrusions is a feature of several IDS solutions. Intrusion prevention systems (IPS) are often used to describe systems that can respond to potential intrusions. The use of specialized tools may also make intrusion detection systems more useful; for example, a honeypot can be used to detect and identify malicious traffic.

Literature Review

Lakshit Sama (2022) [1] Network incursions are becoming more common, complex, and dynamic as network sizes are growing at a fast pace. Identifying breaches in a network of this solving the size issue is a significant and challenging task. Recently, there has been a surge in the use of machine learning for network intrusion detection attention lately because of its smart potential. Machine learning-based solutions, particularly those using deep learning, outperform rule-based solutions when it comes to detecting different types of network attacks. Although deep learning has already shown effective in other domains like image recognition and NLP, network intrusion detection is only starting off. Whether it works in practice is an open question, and if so, there are a number of challenges that must be investigated. Two main difficulties in using deep learning techniques to identify network intrusions is the core focus of this thesis. 1) Inadequate training data in the field of network intrusion detection; 2) Excessive human interaction in current machine learning models; 3) A significant number of false positives and poor accuracy in current deep learning solutions. To deal with insufficient data, we suggest a deep learning system that incorporates LightGBM, XGBoost, LSTM, and decision trees; we test it on three datasets to make sure it works. A few popular datasets are the NSL-KDD, UNSW-NB15, and CIC-IDS2017 databases. A model's likelihood of improving the framework's detection is a key factor in its selection. The proposed framework's results show that all four methods routinely beat the best-in-class machine learning-based solutions, proving that the design methods we used for our thesis were effective.

EBRU KÜLTÜR (2022) [2] Providing security for such systems has become increasingly important due to the fast rise of information technology in several fields. This advancement in IT has also led to a dramatic rise in the sophistication of cyberattacks. This means that older forms of security software, The NSL-KDD, UNSW-NB15, and ACSS datasets are often used. Because they rely on signatures rather than other data, traditional intrusion detection systems can only respond to known threats by using either preset or user-defined rule sets. When new or altered known assaults don't match the static signature, traditional intrusion detection systems miss them. In conjunction with signature-based intrusion detection systems, using deep learning and other developing behavior-based anomalous activity detection approaches might help reduce false positive and negative rates and make new types of attacks easier to spot. Deep learning algorithms may learn from patterns and data, making it viable to assess whether traffic is regular or malicious, thereby increasing the detection rate of actual harmful activity. To cut down on false alarms, they may automate the detecting process automatically, eliminating the need for human setting. This thesis aims to investigate how well network flow-based intrusion detection systems that use deep learning methods, especially recurrent neural network designs, perform.

Miss. Manali Raka *et al*. (2019) [3]. The use of web-based apps has become ubiquitous across all spheres of society, from the private sector to the public sector to the academic world. Most programs are still accessible and operated using web interfaces, even when new technology has emerged. An architecture for multi-stage log analysis is shown in this study. It integrates machine learning techniques in order to match patterns. In order to detect assaults and help stop them, it utilizes logs produced by the application during assaults. A proof-of-concept prototype is built utilizing pattern matching and Bayes Net for machine learning, and the architecture is presented in depth. According to the results of the experiments, the detection accuracy has been enhanced by combining the two systems into a two-stage system. The multi-stage log analysis technique is very relevant to various intrusion detection applications, and the suggested work is crucial in developing online security. In order to identify different kinds of attacks that impact net banking applications, the suggested system makes use of intrusion detection technologies.

Patrick Vanin *et al*. (2022) [4]. The exponential growth of the Internet and associated communications has caused a meteoric rise in the amount of data sent. Due to their insatiable need for this data, attackers are always devising new methods to steal or manipulate it. The rising prevalence of these attacks poses a serious risk to the security of our systems and is one of the main obstacles faced by intrusion detection systems. By analyzing data sent over a network, intrusion detection systems (IDS) may spot malicious activity. A number of academics have looked at intrusion detection system (IDS) issues and come up with new solutions, but both the detection accuracy and the false alarm rate may be better. Moreover, zero-day assaults are difficult for many IDS to identify. In order machine learning techniques have lately been well-liked by scholars for their ability to detect network intrusions with high accuracy and efficiency. Presented below is a taxonomy of ML techniques in this study together with the idea of IDS.

Gabriel c. Fernández, b.s. *et al*. (2019) [5] Organizations need to find innovative solutions to protect their networks and data from ever-changing threat actors, because cyberattacks are becoming more common and more damaging at an exponential pace. It is becoming increasingly difficult to locate a specific security incident or alert due to the exponential growth in the quantity of data

produced by security tools and sensors integrated into today's business networks. When it comes to monitoring, preventing, detecting, and responding to cybersecurity incidents and possible assaults on their networks, organizations must depend on innovative tools to aid and supplement human analysts.

Research Methodology

To determine whether incoming communication is benign or malicious, our proposed method in this study employs machine learning methods. A bespoke business chat web app running on a distant MySQL server forms the backbone of the system. There are two locations where data is recorded: first, in the HTTP traffic that flows between the servers that generate traffic and those that host the web applications. Secondly, in the data transfer between the webapp server and the offline database server using MySQL. Following processing and correlation, a new dataset is generated that incorporates information from both datasets. For our machine learning tasks, we rely on the Weka Machine Learning Framework experiments and compare different algorithms' classification accuracies and efficiency using 5-fold cross-validation throughout the periods of training and while developing the models. Our system adheres to a standard pattern,

Architecture

The four server nodes that make up our design are virtual computers running on an HP server using the KVM technology that has 64G of RAM and two quad-core CPUs. Webapp, traffic generating, database, and Datiphy MySQL data capturing nodes are all part of this network. Figure 1 shows them, and we'll go over them quickly below.

Web Application Server

The webspace is where the bespoke web app is set up., and the webapp server is running Ubuntu/Apache. Located on the database server, SQL Server is the database management system that powers this website. We have two points of data acquisition, and this server is running Snort as well.

Traffic Generation Server

This machine, which runs the Kali Linux system, is used to produce both benign and malevolent traffic. Python and shell programs using the Beautiful Soup Python modules are used to produce both legitimate and malicious traffic.

Database Server

You may run Ubuntu and MySQL on this server. The webapp's chat application connects to this server remotely for database access, and these two servers are the conduits via which the webapp's MySQL traffic travels.

Datiphy MySQL Data Capture Server

This server is comprised of a virtual machine (VM) that Datiphy Inc. has made available for research purposes. For this project, we are making use of an appliance that provides insight into SQL traffic and other forms of database activity. By routing all traffic via this Datiphy appliance, all communication View the connection between the webapp and the MySQL database server using the Datiphy web interface.

Attacks

Injecting SQL functions generally, timing-based assaults, and many kinds of attacks that rely on human mistake are among the manually programmed SQL injection techniques we're utilizing for this project. Sometimes, timing-based attacks are used when the outcomes of injection attempts take a while to become apparent.. Blind SQL injection is another name for these types of attacks. They often look like:

SUBSTRING (version (),1,1) CASE if it's not 4, then sleep (5); if it is, sleep(0)...

An additional five seconds will be added to the request processing time when entering code like this when the first character of the MySQL version is '5'.. Due to the fact that our design uses MySQL version 5.5, this query will wait five seconds before responding, giving an attacker insight into the web application's database backend.

We are using error-based attacks, such as injecting the following code, as an example:... or updatexml (1, concat (0x7e, (user ())),0) or...

An XPATH problem with the database, which might include sensitive information that an attacker could exploit to their advantage (such as the username the webapp is using to access the database), is produced when this code is inserted into a SQL instruction. When working with XML documents in MySQL, you may use XPath, which stands for XMP Path Language. In these assaults, we are using two common procedures in XPATH SQL injections: UpdateXM0L() and Extract Value().

We also use attacks that insert SQL functions inside the INSERT statement, for example, user(), database(), version (), etc. To do this, a single quotation character is used to separate the statement, the function and the characters needed to complete the query are inserted, and finally, characters are used to comment out the remaining portion of the query. Injecting the following code is an example of this kind of attack:', (select database ()), now ()); --

The real input field will take the place of the database name when this code is inserted into an INSERT statement.

Experiment and Results

Summarizes the results of our research with several machine learning techniques. The time it takes to construct the machine learning models is called Model Time, the time it takes to classify the testing dataset using 5-fold cross-validation is called Testing Time, and Accuracy is the classification accuracy that each method achieves.

Since the correlated dataset is a mixture of two datasets, it contains more data than the individual datasets, which, according to my intuitive understanding of the procedure, should provide superior classification accuracy results. It has been the case so far in our research; as previously said, JRip is one example that uses characteristics from both databases to generate rules. Take the JRip algorithm's top-ranking criteria as an example; they combine two datasets-the webapp dataset for http. content length and the Datiphy dataset for SQL Length.

We benchmarked the JRip and J48 algorithms using datasets consisting of 2000 records in our original findings. All of these findings are consistent with the idea that the linked datasets provide much better outcomes than each of the separate datasets in isolation. As a consequence, we can say

that bigger datasets consistently provide better outcomes across all measures. We do use randomness in our attack creation; however, it seems like the data is fitting into categories that the machine learning algorithms can use to build a model. Their ability to do so becomes better with more data. In this research, we have integrated the SVM, Random Forest, and ANN algorithms and documented their performance.

We found that the decision tree method performed quite well in our tests, which is an important part of the performance findings. It is commonly believed that decision tree algorithms are resource intensive. However, we hypothesized that decision tree algorithms outperform other classification algorithms when the data is typically easy to classify using a small number of carefully chosen features. This hypothesis was tested in the present project.

We found that the quicker methods provide results on the correlated dataset that are almost identical to our Multilayer Perceptron findings, but take much less time to execute, based on our trials. According to one conclusion drawn from these studies, using data from several sources may enhance classification results just as much, if not more so, than using algorithms that may be more accurate.

MQTT Ids Dataset Generation

With their widespread adoption over the last decade, IoT devices are projected to reach 25.1 billion by 2025. Many industries and fields make use of IoT networks; such examples are smart cities, agriculture, healthcare, and supply chain management. Among the many protocols that allow for communication between machines in the Internet of Things, MQTT stands out. With an emphasis on the MQTT protocol, Harsha *et al*. examine the protocols used in IoT networks. The authors' study emphasizes the necessity for specialized intrusion detection systems by identifying the many security vulnerabilities connected with MQTT. Their research reveals that 53396 MQTT devices are accessible to the public. The security risks of MQTT are further examined by Dinculeana and Cheng. Based on their findings, more effective methods of detecting MQTT assaults are required.

There is an immediate need to provide current IoT datasets for IDS use since there is a dearth of statistics that reflect IoT traffic. According to the findings and the author's understanding, no intrusion detection system datasets are currently accessible that include MQTT communication, whether it be benign or malicious.

Additionally, an IoT intrusion detection system (IDS) is required due to the growing reliance on IoT and the fact that existing general-purpose IDSs are inadequate for meeting IoT requirements. However, processing, training, and evaluating classification models need datasets that are readily available. Here we provide an IDS dataset that was created using MQTT. First of its kind, this dataset mimics a MQTT-based network with both benign and malicious traffic, standing in for both general and MQTT-specific cyberattacks.

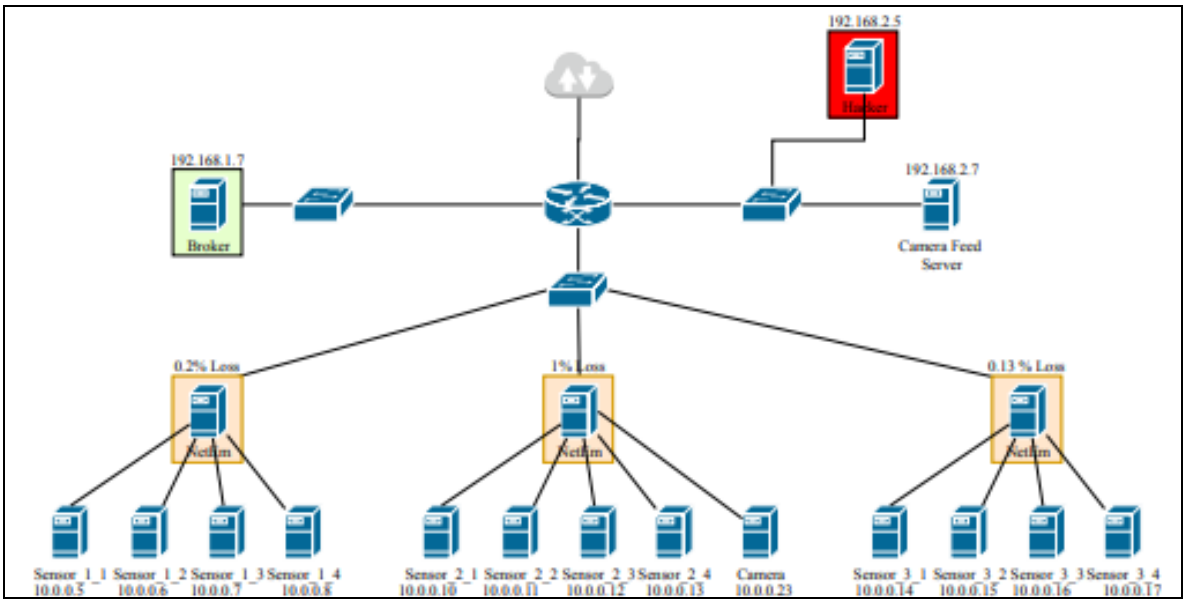


Fig 1: MQTT-IoT-IDS2020: Network Architecture

Scada Dataset Results

Table 1: Sample Confusion Matrix

	Predicted Class				
Correct	**Normal**	**Attack$_1$**	**Attack$_2$**	**Attack$_3$**	**Attack$_4$**
Normal	TN	FB_1	FB_2	FB_3	FB_4
Attack$_1$	FN_1	TP_{11}	TP_{12}	TP_{13}	TP_{14}
Attack$_2$	FN_2	TP_{21}	TP_{22}	TP_{23}	TP_{24}
Attack$_3$	FN_3	TP_{31}	TP_{32}	TP_{33}	TP_{34}
Attack$_4$	FN_4	TP_{41}	TP_{42}	TP_{43}	TP_{44}

With j = 5, the total accuracy is 76.06%, as seen in Table 1. Nonetheless, the classes exhibit either a very high classification accuracy (nearing 100%) or a very poor accuracy (below 50%). Table 2. shows the model accuracy when various j-couples are used for voting. Note that simply labeling all occurrences as the majority class, a ZeroR (baseline majority classifier) will achieve a classification accuracy of 7.14%. Notably, the dataset just includes the register readings from the PLC, and the classes in the raw input space are tightly packed.

Table 2: Siamese Network: SCADA Classification Accuracy Using Different j Votes

No Votes (*j*)	Overall Accuracy	Normal	
		TNR	FPR
1	72.23%	34.3%	65.7%
5	76.06%	48.45%	51.55%
10	77.77%	45.35%	54.65%
15	78.6%	46.65%	53.35%
20	79.18%	47.3%	52.7%
25	79.06%	46.75%	53.25%
25	79.21%	45.7%	54.3%

The use of classification confidence to achieve a greater scenario classification accuracy was motivated by the overlap that was shown in Section 4.4 via the variant classification accuracies. By fitting a k-NN model to the input space data, we can further illustrate that these classes overlap. We choose k=30 to compare it to the Siamese network where 30 pairings are used for majority voting. It should be mentioned that the Siamese network similarity determines the distance between the twin networks' outputs, while the k-NN model determines the distance between instances in their input space. For instance, the k-NN and Siamese networks both mislabeled "S6: 2 Floating Objects" occurrences as seven and three, respectively. Siamese network model has mastered a transformation that lessens class overlap, which explains why S5: Humidity and S8-S10: Person Hitting performed better than before.

Also, the classifications that don't overlap, like S2: Blocked Measures, S3: DoS, and S4: Other, have very good accuracy when it comes to classifying things. Lastly, the Siamese network may decrease the amount of misclassified classes caused by overlap for classes that overlap with more than seven additional classes. On the other hand, these classes were not accurately classified using only "pair similarity."

Conclusion

Intrusion Detection Systems (IDS) are programs that scan every incoming and outgoing data packets for signs of malicious activity. In the last ten years, IDS have been built using a variety of ML approaches. The rapid emergence of new cyber threats has led to the widespread usage of ML approaches. This thesis delves into the use of ML approaches to construct IDS with specific purposes. In addition, this thesis explores the possibility of enhancing the efficiency and efficacy of next-generation IDS by using innovative DL approaches that have been successfully used in other areas of study. Case studies are given here, one for an IoT network and one for a SCADA network, that demonstrate the use of ML approaches to construct specialized intrusion detection systems. An array of six ML methods-LR, NB, k-NN, SVM, DT, and RF-are used. One dataset mimics a control interface (CI) used in a water SCADA system. The fourteen real-life cases included in the collection address topics such as cyber assaults, sabotage, and hardware failures. We do three separate experiments. Binary classification (i.e., normal vs anomalous classes) is executed in the first experiment. The second test determines which part was impacted-none in the event of regular operation, and the part impacted by an anomaly or cyber assault-through classification. The third experiment seeks to categorize the situation that has occurred.

References

1. Sama L. Network intrusion detection using deep learning [thesis]. Melbourne (AU): Victoria University; c2022.
2. Kültür E. Network intrusion detection with a deep learning approach [thesis]. Ankara (TR): [university not specified]; c2022. p. 99.
3. Raka M, *et al*. Intrusion detection using machine learning and log analysis. International Journal of Advanced Research in Computer and Communication Engineering. 2019;8(4).
4. Vanin P, Newe T, Dhirani LL, O'Connell E, O'Shea D, Lee B, *et al*. A study of network intrusion detection systems using artificial intelligence/machine learning. Applied Sciences. 2022;12(22):11752. DOI: 10.3390/app122211752.
5. Fernández GC. Deep learning approaches for network intrusion detection [thesis]; c2019.

Creative Commons (CC) License
This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY 4.0) license. This license permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

IARF Conferences

International Conference on Advances in Science, Engineering, Management and Humanities (ICASEMH- 2023)

ICASEMH/2023/C0223251

CERTIFICATE

This is to Certify that Amarnath Chadchankar, Research Scholar, Department of Computer Science and Engineering, P.K. University, Shivpuri, M.P., India.

has presented a paper on Intelligent Intrusion Detection Models for SQL Database Security Using Machine Learning.

at **International Conference on Advances in Science, Engineering, Management and Humanities (ICASEMH - 2023)** organized by the **International Academic Research Forum (IARF)** at Hotel Siddhartha Inn, Gandhibagh, Nagpur, Maharashtra, India. On 26th February, 2023

Dr. Rayess Ahmad Dar
Organizing Secretary

Dr. Shabnam Arora
Convener

INTERNATIONAL CONFERENCE ON ADVANCES IN SCIENCE, ENGINEERING, MANAGEMENT AND HUMANITIES (ICASEMH – 2023)

26TH FEBRUARY, 2023

CERTIFICATE NO : ICASEMH /2023/C0223251

Intelligent Intrusion Detection Models for SQL Database Security Using Machine Learning

Amarnath Chadchankar

Research Scholar, Department of Computer Science and Engineering,
P.K University, Shivpuri, M.P., India.

ABSTRACT

Machine learning–based intelligent intrusion detection models have emerged as a highly effective solution for strengthening SQL database security in modern digital environments. As organizations increasingly rely on SQL databases to store sensitive information, threats such as SQL injection, data manipulation, privilege escalation, and unauthorized access have become more sophisticated. Traditional rule-based security mechanisms often fail to identify new or evolving attacks, making adaptive and intelligent systems essential. Machine learning models—such as decision trees, random forests, support vector machines, and deep learning networks—analyze large volumes of database logs, query patterns, and user behaviors to automatically identify anomalies that may indicate malicious activity. These models continuously learn from historical attack data and legitimate usage patterns, improving their accuracy in detecting subtle deviations. Intelligent intrusion detection systems not only enhance real-time monitoring but also reduce false alarms by distinguishing between normal and suspicious SQL operations. Features like anomaly detection, pattern recognition, and predictive analytics allow these systems to proactively prevent potential breaches before significant damage occurs. As cyber threats grow more complex, integrating machine learning into SQL database security provides a scalable, automated, and resilient defense mechanism, ensuring data integrity, confidentiality, and overall organizational security.

https://iairconferences.com

IAIR CONFERENCES

National Conference on Latest Innovations in Engineering, Science, Management and Humanities (NCLIESMH– 2024)

NCLIESMH/2024/C0524573

CERTIFICATE

This is to certify that Amarnath Chadchankar,
Research scholar, Department of computer science and Engineering,
P.K. University, Shivpuri, M.P., India.

presented a paper in the National Conference on Latest Innovations in Engineering, Science Management and Humanities (NCLIESMH- 2024)

at The Downton Hotel, Raipur, Chhattisgarh, India on 26th May 2024

Title of the Paper A Study of Efficient Detection of SQL - Based Attacks Through Machine Learning.

K. Jayakumar.
Dr. K. Jaya Kumar
Session Chair

IAIR CONFERENCES IAIR

A Usha Ruby
Dr. A. Usha Ruby
Session Chair

National Conference on Latest Innovations in Engineering, Science, Management and Humanities (NCLIESMH – 2024)

26th May, 2024, Raipur, Chhattisgarh, India.

CERTIFICATE NO : NCLIESMH /2024/C0524573

A Study of Efficient Detection of SQL - Based Attacks Through Machine Learning

Amarnath Chadchankar

Research Scholar, Department of Computer Science and Engineering,
P.K University, Shivpuri, M.P., India.

ABSTRACT

The efficient detection of SQL-based attacks through machine learning represents a transformative approach to modern database security. As SQL databases remain central to organizational data management, they are frequent targets of attacks such as SQL injection, unauthorized data manipulation, and privilege abuse. Traditional security approaches often rely on static rules or signature-based methods that struggle to detect newly emerging threats or subtle anomalies. Machine learning provides a dynamic and adaptive solution by learning patterns of legitimate database use and distinguishing them from malicious behaviors. By analyzing query structures, user activity logs, access frequencies, and behavioral deviations, machine learning models—such as neural networks, naïve Bayes classifiers, random forests, and clustering algorithms—can efficiently identify suspicious activities in real time. These models continuously improve with exposure to new data, allowing them to adapt to evolving attack techniques. Moreover, machine learning reduces the burden of manual monitoring and significantly lowers false positives, enabling security teams to focus on high-risk incidents. The use of predictive analytics further strengthens the ability to anticipate and mitigate threats before they compromise the system. Thus, machine learning offers an efficient, scalable, and proactive framework for safeguarding SQL databases against increasingly sophisticated cyberattacks.