

Enhanced Abnormal Traffic Detection Using Lightweight DAE-GAN and Knowledge Distillation Techniques

Submitted By

Manan Patel

22MCES12



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
INSTITUTE OF TECHNOLOGY
NIRMA UNIVERSITY
AHMEDABAD-382481**

May 2024

Enhanced Abnormal Traffic Detection Using Lightweight DAE-GAN and Knowledge Distillation Techniques

Major Project - II

Submitted in partial fulfillment of the requirements

for the degree of

Master of Technology in Computer Science and Engineering (Cyber Security)

Submitted By

Manan Patel

(22MCES12)

Guided By

Sharada Valiveti



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

INSTITUTE OF TECHNOLOGY

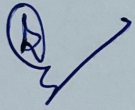
NIRMA UNIVERSITY

AHMEDABAD-382481

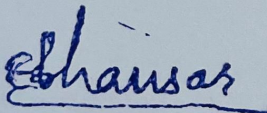
May 2024

Certificate

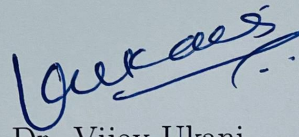
This is to certify that the major project entitled "Advancements in Malware Analysis: Detection and Classification" submitted by Manan Patel (Roll No: 22MCES12), towards the partial fulfillment of the requirements for the award of degree of Master of Technology in Computer Science and Engineering (Cyber Security) of Nirma University, Ahmedabad, is the record of work carried out by him under my supervision and guidance. In my opinion, the submitted work has reached the level required for being accepted for examination. The results embodied in this major project part I, to the best of my knowledge, haven't been submitted to any other university or institution for the award of any degree or diploma.



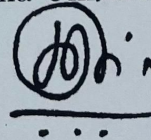
Sharada Valiveti
Guide & Sr. Associate Professor,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.



Dr. Madhuri Bhavsar
Professor and Head,
CSE Department,
Institute of Technology,
Nirma University, Ahmedabad.



Dr. Vijay Ukani
Associate Professor,
Coordinator M.Tech - CSE (Specialization)
Institute of Technology,
Nirma University, Ahmedabad



Dr. Himanshu Soni
Director,
School of Technology,
Nirma University, Ahmedabad

Statement of Originality

I, **Manan Patel**, Roll. No. **22MCES12**, give undertaking that the Major Project entitled "**Enhanced Abnormal Traffic Detection Using Lightweight DAE-GAN and Knowledge Distillation Techniques**" submitted by me, towards the partial fulfillment of the requirements for the degree of Master of Technology in **Computer Science and Engineering (Cyber Security)** of Institute of Technology, Nirma University, Ahmedabad, contains no material that has been awarded for any degree or diploma in any university or school in any territory to the best of my knowledge. It is the original work carried out by me and I give assurance that no attempt of plagiarism has been made. It contains no material that is previously published or written, except where reference has been made. I understand that in the event of any similarity found subsequently with any published work or any dissertation work elsewhere; it will result in severe disciplinary action.

M. R. Patel

Signature of Student

Date: 20 May 2024

Place: Ahmedabad



Endorsed by

Sharada Valiveti

(Signature of Guide)

Acknowledgements

This project has been shaped and enriched by the dedication and guidance of several esteemed individuals. First and foremost, I extend my sincere appreciation to **Sharada Valiveti**, Sr. Associate Professor , Computer Engineering Department, Institute of Technology, Nirma University, Ahmedabad. His unwavering support and the provision of essential facilities have been invaluable to our project's progress.

Special gratitude is owed to **Dr. Madhuri Bhavsar**, Head of Computer Science and Engineering Department, Institute of Technology, Nirma University, Ahmedabad, for her kind support and providing a basic infrastructure and a healthy research environment.

A heartfelt thank you is expressed wholeheartedly to **Dr. Himanshu Soni**, Hon'ble Director, Institute of Technology, Nirma University, Ahmedabad, for the unmentionable motivation he has extended throughout this work.

I would also like to express my gratitude to the entire institution and all faculty members of the Computer Engineering Department at Nirma University, Ahmedabad, for their special attention and suggestions towards the project work.

Further, I acknowledge the authors of the references and other literature consulted during this project, whose works have been instrumental in shaping our research. Lastly, I am immensely grateful to my friends and family. Their encouragement and belief in me have been a pillar of strength throughout this journey.

- **Manan Patel**
22MCES12

Abstract

This thesis addresses the need for efficient and accurate abnormal traffic detection in network security by developing a lightweight Denoise Autoencoder-Generative Adversarial Network (DAE-GAN) model. The original DAE-GAN, known for its high performance, presents significant computational and memory challenges, making it unsuitable for resource-constrained environments. To address this, we employed knowledge distillation, transferring knowledge from a large, complex teacher model to a smaller, efficient student model. We began by implementing the original DAE-GAN, combining a Generative Adversarial Network (GAN) and an autoencoder to identify abnormal traffic. The GAN generated pseudo-anomalies to enhance training, while the autoencoder detected deviations indicative of anomalies. After establishing baseline performance, we applied knowledge distillation to create a lightweight version, training the teacher model on the NSL-KDD dataset and using its soft outputs to guide the student model.

The lightweight DAE-GAN was rigorously evaluated against the original model using the same dataset. Despite reduced complexity and size, it achieved strong performance with 90% accuracy, 85% precision, 97% recall, and a 91% F1 score. These results are competitive with the original DAE-GAN, demonstrating the effectiveness of knowledge distillation in preserving anomaly detection capabilities. This research contributes to the field by providing a practical solution for real-time network intrusion detection on low-resource devices. The lightweight model maintains high detection accuracy while reducing resource consumption, making it suitable for real-world deployment. Future work will focus on deploying the model in live network environments to validate its effectiveness across diverse datasets, advancing resource-efficient machine learning models for cybersecurity applications.

Contents

Certificate	iii
Statement of Originality	iv
Acknowledgements	v
Abstract	vi
List of Figures	ix
1 Introduction	1
1.1 Overview of DAE-GAN and Knowledge Distillation	2
1.2 Objectives of the Research	3
1.3 Thesis Organization	3
2 Literature Survey	5
2.1 Literature Review of Existing Methods	6
2.2 Findings from Literature Survey	12
2.3 Challenges and Gaps in Existing Methods	12
3 Methodology	14
3.1 Base Paper Methodology	14
3.1.1 Model Architecture	15
3.1.2 Advantages	16
3.1.3 Disadvantages	17
3.2 Knowledge Distillation Techniques	17
3.2.1 Concept of Knowledge Distillation	17
3.2.2 Components of Knowledge Distillation	18
3.2.3 The Distillation Process	18
3.2.4 Advantages of Knowledge Distillation	20
3.2.5 Challenges	20
3.3 Development of Lightweight DAE-GAN	20
3.3.1 Introduction	20
3.3.2 Teacher Model: Original DAE-GAN	21
3.3.3 Design of the Student Model	22
3.3.4 Architectural Changes to Simplify the Model	24
3.4 Experimental Setup	25

4	Implementation	26
4.1	Dataset Description	26
4.1.1	Preprocessing Steps	29
4.2	Model Architecture	33
4.2.1	Original DAE-GAN Model Architecture	34
4.2.2	Knowledge Distillation Technique	39
4.2.3	Lightweight DAE-GAN Model Architecture	40
4.2.4	Comparative Analysis	42
4.3	Model Training	42
4.3.1	Training the Original DAE-GAN Model	42
4.3.2	Training the Student Model using Knowledge Distillation	44
4.4	Evaluation Metrics	44
4.4.1	Accuracy	45
4.4.2	Precision	45
4.4.3	Recall	45
4.4.4	F1 Score	46
4.4.5	Confusion Matrix	46
4.4.6	Execution Time	46
4.4.7	Resource Utilization	46
4.4.8	Experiments	47
5	Results and Discussion	49
5.1	Performance Comparison with Base DAE-GAN	49
5.2	Analysis of Lightweight DAE-GAN Results	50
5.3	Impact of Knowledge Distillation	51
5.4	Discussion	53
6	Conclusion	54
6.1	Summary of Findings	54
6.2	Contributions to the Field	55
6.3	Limitations of the Research	55
6.4	Future Work and Recommendations	56
	Bibliography	58

List of Figures

3.1	Knowledge distillation Process	19
4.1	Features of NSL-KDD-dataset	28
4.2	PreProcessing steps	30
4.3	Correlation of Features with label	31
4.4	Model Training Algorithm	37
4.5	Threshold Selection Algorithm	38
4.6	State vectors are taken out of collected packets by the feature extractor and fed into the discriminator. To determine whether the flow is abnormal, the discriminator's detections of abnormal states are added together . . .	48
5.1	results	52

Chapter 1

Introduction

In an era where digital technology underpins nearly every aspect of our personal and professional lives, the threat of malicious software, or malware, looms large. Cyber attackers continuously refine their techniques to evade detection, posing significant challenges to the security of computer systems worldwide. Traditional malware detection methods, though effective to some extent, often struggle to keep pace with the evolving landscape of cyber threats. This has necessitated the development of more advanced and efficient techniques for detecting and mitigating malware.

Malware analysis, a critical component of cybersecurity, aims to understand the behavior and functionality of malware to develop effective countermeasures. Two primary approaches dominate this field: static analysis, which examines malware code without execution, and dynamic analysis, which observes malware behavior in a controlled environment. However, the increasing complexity of malware demands more sophisticated analytical methods.

One promising area of research in malware analysis is the detection of abnormal network traffic. Abnormal traffic often serves as an indicator of malware activity, making it a vital focus for enhancing cybersecurity measures. Among the advanced techniques for abnormal traffic detection, Denoise Autoencoder Generative Adversarial Networks (DAE-GAN) have shown considerable potential. These models leverage the power of deep learning to identify patterns and anomalies in network traffic, contributing to more accurate and efficient malware detection.

However, deploying such sophisticated models on low-resource devices remains a challenge due to their computational and memory demands. To address this issue, this

research focuses on optimizing the DAE-GAN model using knowledge distillation techniques. Knowledge distillation involves transferring the knowledge from a complex, resource-intensive model (the teacher) to a simpler, more efficient model (the student) without significant loss in performance. This approach aims to create a lightweight version of the DAE-GAN model that can perform effectively on low-resource devices, thereby broadening the applicability of advanced abnormal traffic detection methods.

1.1 Overview of DAE-GAN and Knowledge Distillation

The Denoise Autoencoder Generative Adversarial Network (DAE-GAN) is a model developed to enhance abnormal traffic detection, addressing the limitations of traditional detection methods. It combines multiple denoising autoencoders with a discriminator in a GAN-like framework. The denoising autoencoders generate pseudoanomalies, which are then used by the discriminator to learn to distinguish between normal and abnormal traffic. This approach allows the model to transform anomaly detection into a binary classification problem, significantly improving detection sensitivity and precision. The model also includes an efficient real-time feature extraction framework that captures essential spatial and temporal features, ensuring low latency and high accuracy in traffic processing. Evaluations show that DAE-GAN outperforms baseline methods, achieving high precision and recall across various datasets.

Knowledge distillation is a model compression technique that transfers knowledge from a large, complex model (the teacher) to a smaller, more efficient model (the student), enabling the development of lightweight models that maintain high performance and are suitable for resource-constrained environments. In this research, the DAE-GAN serves as the teacher model, providing a high-accuracy benchmark for the student model to emulate. The student model, designed to be simpler and more efficient, is trained to replicate the behavior of the teacher model by minimizing the difference between their outputs, often using the softened output probabilities of the teacher model as additional supervision signals. The distillation process involves training the student model with a combination of the original training data and the knowledge from the teacher model, allowing the student model to learn the essential patterns and features acquired by the teacher, resulting in a compact yet highly effective model.

1.2 Objectives of the Research

The primary objective of Enhanced Abnormal Traffic Detection Using Lightweight DAE-GAN and Knowledge Distillation Techniques. This involves understanding the architecture and functionality of DAE-GAN and replicating its setup to accurately identify abnormal network traffic patterns, which are often indicative of malware activity. By thoroughly evaluating the performance of the original DAE-GAN model, this research aims to establish a benchmark for accuracy, detection rate, and computational efficiency.

Building on the initial implementation, the next objective is to apply knowledge distillation techniques to the DAE-GAN model. Knowledge distillation involves transferring the learned knowledge from a large, complex model (teacher) to a smaller, more efficient model (student). This process ensures that the distilled model retains the critical features and performance characteristics of the original, despite its reduced size and complexity. The application of knowledge distillation is crucial for optimizing the DAE-GAN model, making it more suitable for deployment in environments with limited computational resources.

The final objective is to develop a lightweight version of the DAE-GAN model that achieves comparable results to the original model. This involves designing and training a streamlined model that maintains high detection accuracy and robustness while significantly reducing its computational and memory requirements. Extensive testing and evaluation will be conducted to compare the performance of the lightweight model against the original DAE-GAN, ensuring that the efficiency gains do not come at the expense of detection capabilities. The successful realization of this objective will demonstrate the feasibility of deploying advanced abnormal traffic detection models on low-resource devices, enhancing their practical applicability in real-world cybersecurity scenarios.

1.3 Thesis Organization

This thesis is organized into seven chapters: Chapter 1 introduces the importance of malware analysis and the objectives of the research. Chapter 2 reviews relevant literature on malware analysis, traffic identification, and abnormal traffic detection methods, focusing on DAE-GAN and knowledge distillation. Chapter 3 details the research methodology, including the implementation of DAE-GAN and the application of knowledge distillation

to develop a lightweight model. Chapter 4 covers the practical implementation, including dataset, model architecture, and training process. Chapter 5 presents and discusses the experimental results, comparing the performance of the original and lightweight DAE-GAN models. Chapter 6 summarizes the findings, contributions, limitations, and suggests future research directions. Chapter 7 lists all references cited in the thesis.

Chapter 2

Literature Survey

The identification and prevention of malicious software, also known as malware, is a vital part of cybersecurity. Many different techniques have been created to fight against the constantly changing nature of malware.^[1] These methods fall into three primary categories: signature-based, heuristic, and anomaly-based methods.

1. **Signature-Based Detection:**Signature-based malware detection is an old and established technique that uses predefined signatures or patterns of known malware to identify malicious software. Security software stores a database of these signatures, which are unique characteristics or fingerprints of known malware types. When a file is scanned, the software compares it with the signature database, and if there is a match, the file is marked as malware. This method is efficient, accurate, and quick at identifying known malware strains. However, it has a significant downside. It cannot detect new or unknown malware since it depends on existing signatures. Malware developers are continually updating their code to evade detection, making signature-based techniques less effective against zero-day threats.
2. **Heuristic Detection:**Heuristic malware detection takes a more dynamic approach, looking for suspicious behaviors and characteristics that may indicate malware. If a program exhibits suspicious activities like modifying system files or creating unauthorized network connections, it is flagged as potentially malicious. Some heuristic methods run suspicious files in a controlled environment (sandbox) to observe their behavior without risking the host system. Heuristic detection is more capable of identifying previously unknown malware since it doesn't rely on signatures. However,

it can produce false positives, flagging legitimate software with certain behaviors as malicious.

3. **Anomaly-Based Detection:** It is a departure from the traditional methods. Instead of looking for known malware signatures or heuristic indicators, it focuses on identifying anomalies in system behavior or data patterns. Systems first establish a baseline of normal behavior for a system or network. This baseline is learned from historical data. The system continuously monitors the system or network for deviations from the established baseline. Any significant deviation is considered suspicious and may indicate malware. Many anomaly-based systems use machine learning algorithms to identify these deviations. These algorithms can adapt and improve their detection capabilities over time. Anomaly-based detection is effective at identifying previously unknown malware and zero-day threats since it doesn't rely on known patterns. However, it can produce false positives if the baseline is not well-established or if the system undergoes significant legitimate changes.

2.1 Literature Review of Existing Methods

[1] Use efficient feature extraction and a unique DAE-GAN model to reduce false alarms. By generating pseudoanomalies through DAE-GAN, they achieve high precision rates of 98.6% on NSL-KDD. Their approach excels in flow-wise precision (over 99%) and has a respectable recall of 60.6% on four attack datasets.

[2] generating synthetic zero-day attack data using GANs, emphasizing its significance in cybersecurity. By creating a sizable dataset of zero-day attacks, the study demonstrates GANs' ability to enhance the training of DL classifiers. The conclusion highlights the improved performance of a Neural Network trained on a combination of original and generated data, offering a valuable solution to the scarcity of large, high-quality datasets for zero-day threat detection.

[3] introduces ARCADE, an unsupervised learning system for detection of anomaly in network. ARCADE automatically creates normal traffic profiles from raw network data and uses adversarial regularization to enhance anomaly detection. It achieves nearly 100% F1-scores in detecting malware and network attacks, even with just two initial packets of data. ARCADE is also much smaller in size, faster, and more accurate than existing methods.

[4] A novel approach to multi-domain machine learning for detecting coordinated network attacks, using an attention-based Bi-LSTM model. It effectively handles different attack domains by treating HTTP traffic as structured natural language sequences. Experimental results show strong performance in detecting various network anomalies, including malware, VPN encapsulation, and Trojan horses.

[5] This paper evaluates classification algorithms for network anomaly detection using the UNSW-NB15 dataset. It optimizes these algorithms with various encoding methods and data ratios, highlighting the Random Forest Classifier as the top performer with an F2-score of 97.68% and an AUC score of 98.47%. It introduces efficient label encoding and feature reduction techniques for NetFlow data streams, making it a valuable contribution to network anomaly detection research.

[6] introduces ZeVigilante, a system for detecting of Zero-Day attack using machine learning and Cuckoo sandboxing. ZeVigilante achieves high accuracy, with Random Forest leading at 98.21% for static analysis and 98.92% for dynamic analysis. It outperforms existing methods and holds promise for Zero-Day malware detection. Future work may focus on classification improvement, memory optimization, and sustainability evaluation of ML-based malware detectors.

[7] Aims to improve zero-day malware detection using machine learning models, specifically bagging and boosting. By leveraging Shapley values of features transformed into a probability scale, the study enhances model performance. It identifies XGBoost as the best-performing model, achieving high accuracy. The paper proposes a method for detecting and rectifying misclassifications in false negatives and false positives through the analysis of top features using waterfall plots.

[8] Focuses on detecting zero-day malware, previously unknown software vulnerabilities, using the PlausMal-GAN framework. PlausMal-GAN generates analogous zero-day malware data for training and detection. The framework outperforms various GAN models and offers an efficient approach to zero-day malware detection without relying on signature analysis. Future research may involve expanding malware types and optimizing GAN models for broader applications.

[9] An unsupervised method for detecting encrypted malware traffic using a three-layer Autoencoder for feature compression and the Kmeans algorithm for classification. It addresses the challenge of efficient detection without requiring extensive labeled data.

Results show competitive performance with an F1-measure of 0.95, similar to supervised methods. This approach is valuable for identifying encrypted malware traffic efficiently.

[10] ADRIoT is an IoT network anomaly detection framework. It uses edge computing to enhance security and employs unsupervised LSTM autoencoders for constructing anomaly detectors capable of handling emerging zero-day attacks. To address resource constraints on edge devices, a multiedge collaborative mechanism is proposed.

[11] a semi-supervised method for zero-day malware detection, combining autoencoding and one-class classification to leverage neural networks and simplify threshold selection. Experimental results show up to 97.1% accuracy, with resilience to adversarial attacks. Future research includes exploring defense mechanisms against attacks and applying the method to various applications beyond malware detection.

Paper	Research Objective	Scope	Dataset	Key Findings	Application
[1]	To develop a method for abnormal traffic detection using machine learning techniques.	Abnormal traffic detection in real-world applications, considering the need for real-time performance and the limitations of computational complexity	Kitsune, NSL-KDD, UNSW-NB15	Proposed a novel approach for abnormal traffic detection combining DAE and GAN.	Traffic management for identifying traffic anomalies, Anomaly detection in industrial systems, Fraud detection in financial transactions
[2]	creating data about zero-day attacks and applying deep learning and machine learning approaches to enhance their detection.	adoption of effective machine learning approaches for the detection of zero-day assaults and investigates the use of feature selection methods for data including zero-day malware.	kaggle-data	Demonstrated that GAN-generated zero-day attack data can improve the performance of deep learning classifiers in detecting previously unseen threats.	Improving the effectiveness of intrusion detection systems, Addressing the challenges of zero-day attacks in various domains
[3]	Developing a novel approach (ARCADE) for network anomaly detection using an adversarially regularized convolutional autoencoder.	It introduces a convolutional autoencoder architecture regularized by adversarial training for improved detection of network anomalies.	ISCX-IDS, USTC-TFC, MIRAI-RGU	Proposed ARCADE, an innovative convolutional autoencoder model with adversarial regularization, which enhances the detection of network anomalies.	Intrusion detection in computer networks, Identifying malicious activities in cloud environments.

Paper	Research Objective	Scope	Dataset	Key Findings	Application
[4]	Developing an intrusion detection system using ensemble-bidirectional Long Short-Term Memory (LSTM) networks for improved network security.	Utilization of multi-domain data and ensemble-bidirectional LSTM networks to enhance the detection of network intrusions.	CCCS-CIC-AndMal2020, CTU-13 dataset, ISCX dataset, KDDcup99 dataset	Proposed an ensemble-bidirectional LSTM model for network intrusion detection, which effectively captures temporal dependencies in network traffic data.	Protecting computer networks from external attacks, Securing critical infrastructure and industrial control systems
[5]	boosting the classification of network traffic anomalies' robustness	NetFlow streams and various types of network anomalies	UNSW-NB15	Explored the applicability of supervised machine learning algorithms for NetFlow traffic anomaly detection.	Classification and machine learning-based detection of anomalies in network traffic
[6]	Developing an approach, ZeVigilante, for the detection of zero-day malware using a combination of machine learning and sandboxing analysis techniques.	The paper's scope is centered on cybersecurity and malware detection. It explores the effectiveness of ZeVigilante in identifying previously unknown, zero-day malware threats by integrating machine learning models and sandboxing analysis.	API Call Sequences Dataset, Top 1000 PE Imports	Introduced ZeVigilante, a novel approach for zero-day malware detection.	Protecting end-users and organizations from evolving malware threats.

Paper	Research Objective	Scope	Dataset	Key Findings	Application
[7]	The objective of the research is to detect zero-day malware using machine learning models and visualize the top significant features of malware.	Analysis of top features and their contribution in predicting malware and benign software	P. EMBER	Introduced an ensemble approach that combines Shapley values with boosting and bagging for zero-day malware detection.	Enhancing the security of computer systems and networks against zero-day malware.
[8]	To develop a framework for detecting and classifying zero-day malware using generative adversarial networks (GANs)	It aims to train a discriminator using malware images generated by a generator to detect and classify zero-day malware.	MALIMG	The proposed framework, trained up to phase 2, showed high and stable accuracy in detecting and classifying zero-day malware	Developing zero-day malware detection software
[9]	To develop an unsupervised anomaly detection framework for encrypted malware traffic.	The detection of abnormal traffic in encrypted malware using the TLS protocol	Datacon 2020-encrypted	The performance of the SVM algorithm is shown to degrade with the presence of noisy labels.	Detection of abnormal traffic in encrypted malware, enhancing network security.
[10]	To develop an anomaly detection module for inspecting the traffic of an IoT network and detecting malicious activities.	Working collaboratively to capture, preprocess, and detect anomalies in IoT network traffic, including the generation of alarms for malicious activity detection.	11	Unsupervised methods, like ADRIoT, show advantages over supervised methods in detecting unknown attacks.	It can help enhance the security of IoT devices and networks by identifying and preventing potential attacks.

Paper	Research Objective	Scope	Dataset	Key Findings	Application
[11]	To create and assess malware detection techniques utilizing the Drebin and Meraz'18 datasets.	The goal of the research is to create and assess malware detection techniques utilizing semi-supervised learning based on robust data abstraction.	Meraz'18	CSVM with RBF kernel outperforms other kernels in terms of performance.	Zero-day malware detection and mobile-platform malware detection.

2.2 Findings from Literature Survey

The literature review reveals various methodologies for detecting zero-day attacks through anomaly-based detection techniques, including DAE-GAN, GAN, BI-LSTM, Convolutional Autoencoder, Cuckoo sandboxing, PlausMal-GAN, three-layer Autoencoder, and LSTM autoencoders. These techniques have been evaluated using datasets such as Kitsune, NSL-KDD, UNSW-NB15, ISCX-IDS, USTC-TFC, MIRAI-RGU, Datacon2020-encrypted, and Meraz'18, with performance metrics like accuracy, precision, recall, and F1 score. Among these, DAE-GAN stands out for abnormal traffic detection due to its semisupervised learning capability, computational efficiency, effectiveness in high-dimensional spaces, and its innovative combination of pseudoanomaly and adversarial learning. Additionally, DAE-GAN incorporates a sophisticated feature extraction framework that enhances its ability to detect anomalies accurately and scalably. These findings highlight the superior performance and adaptability of DAE-GAN in the evolving landscape of network security.

2.3 Challenges and Gaps in Existing Methods

Despite the advancements in anomaly-based detection techniques, existing methods such as GAN, BI-LSTM, Convolutional Autoencoder, Cuckoo sandboxing, PlausMal-GAN, three-layer Autoencoder, and LSTM autoencoders have notable limitations. These models often struggle with computational inefficiency, making them impractical for deploy-

ment on low-resource devices. Additionally, many of these methods require extensive labeled data for training, lack scalability in high-dimensional spaces, and exhibit high false-positive rates. The DAE-GAN model, while effective, also shares these constraints due to its complexity and resource-intensive nature. This research identifies a critical gap in the development of lightweight models that maintain high detection accuracy while being suitable for real-time, resource-constrained environments. By applying knowledge distillation to the DAE-GAN, this project aims to create a more efficient, lightweight version of the model that delivers comparable results to the original, addressing the need for scalable and accessible abnormal traffic detection solutions.

Chapter 3

Methodology

3.1 Base Paper Methodology

The Denoise Autoencoder Generative Adversarial Network (DAE-GAN) is an advanced model developed for abnormal traffic detection, addressing the limitations of traditional methods. Traditional abnormal traffic detection techniques can be categorized into misuse-based and anomaly-based approaches. Misuse-based methods, relying on signature matching, struggle with zero-day attacks due to their dependence on predefined signatures, which require continuous updates. Anomaly-based methods detect deviations from normal traffic patterns, offering the advantage of identifying new, unseen attacks. However, these methods often suffer from high false positives and computational inefficiencies, particularly in high-dimensional data spaces. The DAE-GAN model overcomes these challenges by integrating multiple denoising autoencoders with a discriminator in a GAN-like structure. The autoencoders generate pseudoanomalies, which the discriminator learns to distinguish from normal traffic through adversarial training. This approach transforms anomaly detection into a binary classification problem, significantly enhancing the model's sensitivity and precision. Additionally, an innovative real-time feature extraction framework is employed, using a packet window to capture spatial and temporal features, ensuring low latency and high accuracy in traffic processing. The effectiveness of DAE-GAN is demonstrated through extensive evaluations on datasets such as NSL-KDD and UNSW-NB15, where it outperforms baseline methods in precision and recall, proving its robustness and efficiency in both GPU-based platforms and mobile devices.

Though their efficacy varies depending on the algorithm employed, anomaly-based

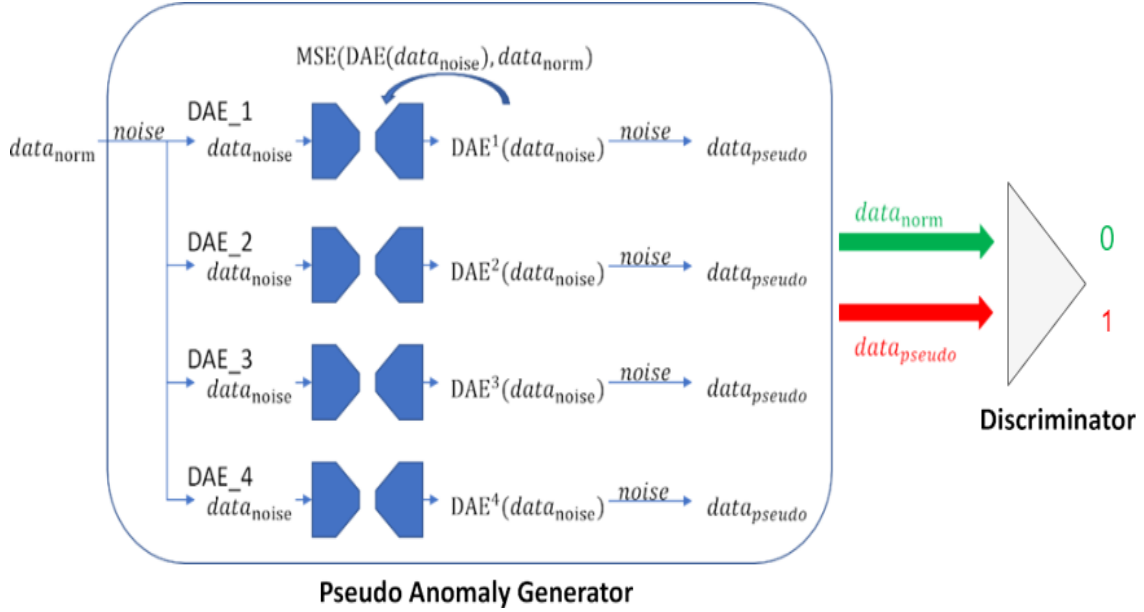
techniques are well known for their capacity to identify hits that have not been seen before. There are three main types of unsupervised anomaly detection methods: reconstruction-based, clustering-based, and one-class classification. All of them are trained on normal data without any supervision. However, their performance can be negatively affected if there are no anomalous patterns in the training data. This can lead to a low rate of detection recall and a high frequency of false alarms even when they are able to detect new anomalies. Practically speaking, this can be addressed by sampling within the feature space to create pseudo-anomalies. Random sampling, however, is not feasible in any data space, particularly for high-dimensional data sets. It takes a more effective data augmentation method to take advantage of pseudo-anomalies. In particular, instead of using actual abnormal traffic (like virus traffic) that is gathered from the Internet, these pseudo-anomalies should be artificially generated to mimic abnormal traffic.

Our approach involves the use of Deep Autoencoders (DAEs) to produce pseudo anomalies effectively. These pseudo anomalies have the most significant impact when they are similar to normal instances. By training the discriminator with these pseudo anomalies, we can create a highly responsive anomaly detector. The anomalies created by DAE-GAN closely resemble the distribution of normal data, which greatly reduces the number of outliers in the generated anomalies. To ensure the trained model’s convergence and effectiveness, we can fine-tune hyperparameters such as the noise factor and the latent dimensionality of the DAEs, depending on the complexity of the training data.

3.1.1 Model Architecture

The DAE-GAN anomaly detection model, shown in the figure, consists of a pseudorandom generator and a discriminator. The PA generator is made up of several denoising autoencoders. The normal data is mixed with noise and processed by these DAEs. The output is then used as pseudo abnormal samples. The reconstruction capability of the DAEs is limited by the size of their latent space and the noise level.

On the right side of the figure, the discriminator acts as a binary classifier. It learns to distinguish between abnormal samples by being trained on a mix of normal samples and those generated by the PA generator. This training process is based on adversarial learning, which involves continuous interaction between the PA generator and the discriminator.



design of DAE-GAN. Original data are tainted by noise or masks in order to train a DAE. By minimizing the loss function applied to the original data, the polluted data are reconstructed in the output, and the reconstructed data can compel the DAE to acquire robust features.

Figure 3 shows the structure of a Deep Autoencoder (DAE). The process begins by adding random noise to the input data. The DAE then focuses on minimizing reconstruction errors to reconstruct the most important features of the original data. In essence, a DAE learns from noisy normal data to identify normal traffic features. Then, it adds specific noise distribution to the normal data, creating pseudoanomalies for training the discriminator. The noise added to the original data is produced by multiplying standard normal random noise $N(0,1)$ with a hyperparameter called the noise factor (f_{noise}). *By adjusting f_{noise} , you can control the similarity between the generated and original samples, which is tuned by tweaking f_{noise} and another parameter, $latent_{dim}$. This flexibility helps prevent the problem of "m*

3.1.2 Advantages

The process involves extracting features from four public datasets and selecting them using a combination of different algorithms such as Information Gain (IG), Information Gain Ratio (IGR), Chi-square (2), and ReliefF. IG, IGR, and 2 are particularly useful in determining correlations among discrete features. On the other hand, ReliefF is excellent at identifying optimal division points for continuous features as it functions as a discretization algorithm.

All four algorithms are applied to each of the datasets. To ensure comprehensive

feature selection, we combine the features selected by each algorithm. Furthermore, the final set of features is an amalgamation of those derived from the four distinct public traffic datasets, enhancing the overall generalizability of the feature sets.

In addition, the flexibility of Deep Autoencoders (DAEs) to adjust their reconstruction capability through tuning of the noise factor (f-noise) and latent dimensionality (latent-dim) is utilized. This approach helps circumvent the issue of 'mode collapse' and increases sensitivity, especially when using multiple DAEs as Pseudoanomaly (PA) generators.

3.1.3 Disadvantages

Our model's discriminator is trained by using anomalies generated by the PA generator. This allows it to distinguish between normal and abnormal internet traffic. Our approach is different from traditional methods that mainly rely on normal data for detecting anomalies. Instead, we merge the idea of pseudo anomalies with adversarial learning in our strategy.

At its core, the system adopts a GAN-like architecture, comprising a proficient PA generator and a binary discriminator. This design effectively redefines anomaly detection as a binary classification challenge. The discriminator's ability to detect anomalies is further enhanced through the process of adversarial learning.

3.2 Knowledge Distillation Techniques

Knowledge distillation is a model compression technique that has garnered significant attention in the field of machine learning, particularly for its ability to transfer knowledge from a larger, complex model (often referred to as the teacher) to a smaller, simpler model (referred to as the student). This process allows the smaller model to achieve a performance level comparable to the larger model while being more efficient in terms of computational resources and memory usage. The technique is especially valuable in scenarios where deploying resource-intensive models is impractical, such as on mobile devices or edge computing environments.[\[12\]](#)

3.2.1 Concept of Knowledge Distillation

The concept of knowledge distillation was introduced by Geoffrey Hinton and his colleagues in 2015. The primary idea is to use the outputs of the teacher model as soft targets to train the student model. Unlike hard targets, which are binary or categorical labels,

soft targets provide a probability distribution over the possible classes. This probability distribution contains richer information about the learned patterns and relationships in the data, which the student model can leverage to improve its own performance.[13]

3.2.2 Components of Knowledge Distillation

Teacher Model: The teacher model is typically a deep and complex neural network that has been trained to achieve high accuracy on a given task. It represents the gold standard that the student model aims to approximate.[14] In the context of this research, the Denoise Autoencoder Generative Adversarial Network (DAE-GAN) serves as the teacher model, excelling in abnormal traffic detection.

Student Model: The student model is designed to be simpler and more efficient than the teacher model. The goal is to achieve similar performance metrics (such as accuracy, precision, recall, and F1 score) while reducing the computational burden. The student model is often a shallow neural network or a model with fewer parameters compared to the teacher.

Soft Targets: During training, the teacher model produces soft targets, which are the probability distributions over the output classes. These soft targets contain more information than hard targets because they reveal not only the predicted class but also the confidence level of the predictions for each class.

Distillation Loss: The distillation process involves a specialized loss function that combines the traditional loss (e.g., cross-entropy loss) with a distillation loss. The distillation loss measures the difference between the soft targets produced by the teacher model and the outputs of the student model. A temperature parameter is often introduced to soften the probability distributions, making it easier for the student model to learn from the teacher.

3.2.3 The Distillation Process

The knowledge distillation process can be broken down into several key steps:

1. **Train the Teacher Model:** The first step is to train the teacher model on the available training data. The teacher model learns to perform the task with high accuracy, generating soft targets as a byproduct.
2. **Generate Soft Targets:** Once the teacher model is trained, it is used to generate

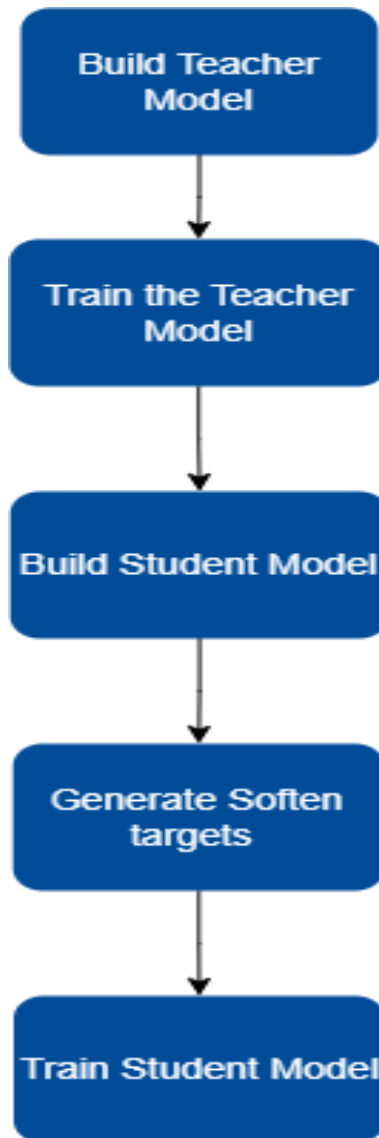


Figure 3.1: Knowledge distillation Process

soft targets for the training data. These soft targets include the probability distribution over the output classes, which provides additional information beyond the hard labels.

3. **Train the Student Model:** The student model is trained using a combination of the original training data and the soft targets produced by the teacher model. The loss function used during this training phase is a combination of the standard loss (e.g., cross-entropy loss with hard labels) and the distillation loss, which measures the discrepancy between the soft targets and the student's predictions.
4. **Temperature Scaling:** A temperature parameter is applied to soften the soft targets. This parameter smooths the probability distribution, making it easier for

the student model to learn from the teacher. The temperature parameter is usually greater than 1 during the distillation process.

3.2.4 Advantages of Knowledge Distillation

- **Model Compression:** Knowledge distillation effectively compresses a large, complex model into a smaller, more efficient model without significant loss of performance. This is particularly useful for deploying machine learning models on devices with limited computational resources.
- **Enhanced Generalization:** The student model benefits from the rich information contained in the soft targets, which can lead to better generalization on unseen data. This is because the student model learns not only the correct labels but also the nuanced relationships between different classes.
- **Improved Training Efficiency:** By leveraging the soft targets from the teacher model, the student model can often achieve high performance with fewer training iterations compared to training from scratch with hard labels alone.
- **Transferability:** Knowledge distillation can be applied to various types of models and tasks. It is a flexible technique that can be used to transfer knowledge from any type of complex model to a simpler one.

3.2.5 Challenges

While knowledge distillation offers significant advantages, it also presents some challenges. One challenge is ensuring that the student model captures all the essential features learned by the teacher, especially when the teacher is significantly more complex. Additionally, selecting the appropriate temperature parameter and balancing the loss components can be non-trivial and may require extensive experimentation.

3.3 Development of Lightweight DAE-GAN

3.3.1 Introduction

The primary purpose of developing a lightweight Denoise Autoencoder Generative Adversarial Network (DAE-GAN) is to create a model that maintains the high performance of the original DAE-GAN while being suitable for deployment on low-resource devices.

The original DAE-GAN model, although highly effective in detecting abnormal traffic patterns, is computationally intensive and resource-demanding, limiting its applicability in environments with constrained computational capabilities, such as mobile devices and edge computing platforms. By leveraging knowledge distillation techniques, this research aims to distill the essential knowledge and capabilities of the complex DAE-GAN model into a more compact and efficient version. The resulting lightweight model should deliver comparable results to the original, ensuring that advanced anomaly detection can be performed efficiently even on devices with limited processing power and memory. This development not only enhances the practical usability of DAE-GAN but also broadens the scope of its application in real-world scenarios where resource constraints are a significant consideration.

3.3.2 Teacher Model: Original DAE-GAN

The original Denoise Autoencoder Generative Adversarial Network (DAE-GAN) model, serving as the teacher in the knowledge distillation process, is composed of two primary components: the Generative Adversarial Network (GAN) and the Autoencoder. These components work together to detect anomalies in network traffic by leveraging the strengths of both generative and discriminative modeling.

GAN Model

The GAN component of the DAE-GAN consists of a generator and a discriminator, which are trained simultaneously through an adversarial process. The generator aims to produce data that mimics the normal traffic patterns, while the discriminator learns to distinguish between real (normal) and generated (pseudoanomalous) traffic. The adversarial training helps the GAN component improve its ability to model the distribution of normal traffic data, enhancing the overall anomaly detection capabilities of the DAE-GAN.

Total Parameters: 77,041 (300.94 KB)

Trainable Parameters: 53,488 (208.94 KB)

Non-trainable Parameters: 23,553 (92.00 KB)

The relatively high number of parameters in the GAN component reflects its complexity and capacity to capture intricate patterns in the network traffic data. The trainable

parameters include weights and biases that are updated during the training process, while the non-trainable parameters typically consist of fixed elements such as pre-trained embeddings or constants.

Autoencoder Model

The Autoencoder component of the DAE-GAN is designed to denoise and reconstruct normal traffic data. It comprises an encoder, which compresses the input data into a latent space representation, and a decoder. The autoencoder's main function is to reconstruct the original data from a compressed representation. Its primary purpose is to learn a concise representation of normal traffic patterns, thereby simplifying the detection of deviations that indicate anomalies.

Total Parameters: 23,466 (91.66 KB)

Trainable Parameters: 23,466 (91.66 KB)

Non-trainable Parameters: 0 (0.00 Byte)

The autoencoder's parameters are all trainable, emphasizing its role in learning and refining the representation of normal traffic data. The relatively lower number of parameters compared to the GAN component indicates a more streamlined architecture focused on efficient encoding and decoding.

Combined Architecture

The integration of the GAN and Autoencoder components in the DAE-GAN model provides a robust framework for abnormal traffic detection. The autoencoder effectively compresses and reconstructs normal traffic data, while the GAN enhances the model's ability to discern between normal and anomalous patterns through adversarial training. This combined approach allows the DAE-GAN to leverage both generative and discriminative strengths, resulting in a powerful tool for identifying anomalies in network traffic.

3.3.3 Design of the Student Model

Creating a lightweight or student model using knowledge distillation involves several design considerations to ensure the model remains efficient while maintaining performance comparable to the original, complex teacher model. The goal is to develop a model that can be deployed on low-resource devices without significant loss of accuracy or functionality. Here are the primary parameters and considerations:

1. Model Architecture Simplification

- Layer Reduction: Reduce the number of layers (e.g., from 10 to 5).
- Neuron Reduction: Decrease neurons per layer (e.g., from 512 to 256).
- Depth and Width: Balance the number of layers and neurons to maintain performance.

2. Parameter Pruning

- Weight Pruning: Remove less important weights to reduce model size and improve efficiency.
- Neuron Pruning: Eliminate neurons that contribute the least to output.

3. Knowledge Distillation Specifics

- Soft Targets: Use teacher model's soft targets to provide richer information for the student model.
- Temperature Parameter: Soften probability distribution to aid student model learning
- Loss Function: Combine traditional loss with distillation loss for effective knowledge transfer.

4. Hardware Efficiency

- Memory Usage: Reduce the model's memory footprint for devices with limited RAM.
- CPU/GPU Usage: Optimize for efficient execution, reducing latency and improving throughput.
- Inference Speed: Ensure quick inference for real-time applications.

Example Reductions

Model Size: Reduce parameters by 50-90%.

Memory Usage: Decrease memory usage by a similar proportion.

CPU Usage: Lower CPU utilization during inference by 50% or more.

Inference Time: Reduce latency by 50-75%.

3.3.4 Architectural Changes to Simplify the Model

The development of the student GAN and autoencoder models involved significant architectural changes aimed at reducing computational complexity while maintaining performance. These modifications were guided by the principles of knowledge distillation, where the goal is to create lightweight models that can be efficiently deployed on low-resource devices.

Student GAN Model The original GAN model was simplified by reducing both the number of layers and the number of neurons per layer. The total parameters were reduced from 77,041 to 16,657. Specifically, the generator and discriminator layers were downsized, with the generator’s dense layers reduced from 256 and 128 neurons to 128 and 64 neurons, respectively. Similarly, the discriminator’s layers were also simplified. This reduction in depth and width helped decrease the model’s complexity significantly. Additionally, parameter pruning was employed to remove less important weights, further minimizing the number of trainable parameters to 13,008, with non-trainable parameters at 3,649. These changes ensure that the student GAN model is not only smaller in size but also more computationally efficient.

Student Autoencoder Model The student autoencoder model was designed to be much more streamlined compared to the original. The total number of parameters was reduced from 23,466 to 6,033. This involved decreasing the number of layers in both the encoder and decoder parts of the autoencoder and reducing the number of neurons in each layer. For example, layers that previously had a large number of neurons were scaled down to fewer neurons while still capturing essential features. The simplification process also included efficient feature representation techniques to maintain the performance of the autoencoder. With all 6,033 parameters being trainable, the student autoencoder model retains its effectiveness in feature extraction while being significantly lighter and faster.

By implementing these architectural changes and leveraging knowledge distillation, the student GAN and autoencoder models were effectively compressed, making them suitable for deployment on low-resource devices without significant loss in performance. This ensures that advanced anomaly detection capabilities can be accessed more broadly, facilitating practical applications in real-world scenarios.

3.4 Experimental Setup

The experimental setup for developing and evaluating the lightweight DAE-GAN model was conducted using the NSL-KDD dataset, which provided a comprehensive set of network traffic data for both training and evaluation. The training process was carried out on Google Colab, leveraging its computational resources to handle the intensive tasks. Key training parameters included a learning rate of 0.002, a batch size of 128, and a total of 50 epochs. The Adam optimizer was employed to ensure efficient and effective convergence during training. This setup facilitated the development of a robust and efficient model, enabling thorough experimentation and performance validation.

Chapter 4

Implementation

In this section, the process and observations of implementation is discussed.

4.1 Dataset Description

In our study, we utilized the NSL-KDD dataset, The NSL-KDD dataset is a refined version of the KDD Cup 1999 dataset, created to address several issues and shortcomings present in the original dataset. It is widely used for evaluating intrusion detection systems and network anomaly detection models. Below is a detailed description of the NSL-KDD dataset, including its structure, features, and relevance to our research.

1. Overview : The NSL-KDD dataset was designed to provide a more realistic and comprehensive benchmark for evaluating intrusion detection systems. It includes various types of network traffic data, categorized into normal and malicious activities. The dataset consists of training and testing sets that are carefully crafted to mitigate the inherent biases and redundant records found in the KDD Cup 1999 dataset.
2. Structure of the Dataset : The NSL-KDD dataset is divided into the following key components:
 - KDDTrain+: The primary training set.
 - KDDTest+: The primary testing set.
 - KDDTrain+_20Percent: A 20% subset of the KDDTrain+ set, often used for faster experimentation.
 - KDDTest-21: A challenging subset of the test set, including some attack types not

present in the training set.

Each record in the dataset represents a network connection and is described by 41 features along with a label indicating whether the connection is normal or belongs to a specific type of attack.

3. **Features :** The NSL-KDD dataset comprises 41 features categorized into three main groups: Basic Features, Content Features, and Traffic Features. Basic Features are extracted directly from packet headers without inspecting the payload and include attributes such as Duration (the length of the connection in seconds), Protocol_type (the type of protocol, such as TCP, UDP, or ICMP), Service (the network service on the destination, such as HTTP or FTP), and Flag (the status flag of the connection). Content Features are derived from the packet payloads, offering insights into the data within a connection. Examples include Src_bytes (the number of data bytes sent from the source to the destination), Dst_bytes (the number of data bytes sent from the destination to the source), and Count (the number of connections to the same host as the current connection within the past two seconds). Traffic Features are calculated using a two-second time window, focusing on connection behaviors such as Same_srv_rate (the percentage of connections to the same service), Serrror_rate (the percentage of connections with SYN errors), and Srv_serrror_rate (the percentage of connections to the same service with SYN errors). These features collectively provide a comprehensive understanding of network traffic patterns, aiding in the detection of anomalies..
4. **Attack Types :** The dataset includes four main categories of attacks, each representing different malicious behaviors: DoS (Denial of Service) attacks aim to make a machine or network resource unavailable to its intended users, with examples including smurf, neptune, and back. R2L (Remote to Local) attacks involve unauthorized access from a remote machine, exemplified by guess_passwd, ftp_write, and imap. U2R (User to Root) attacks allow the attacker to gain root access to the system, with examples such as buffer_overflow, rootkit, and perl. Probe attacks involve the surveillance and probing of machines to gather information or find vulnerabilities, including examples like satan, ipsweep, and nmap.
5. **Relevance to Abnormal Traffic Detection :** The NSL-KDD dataset is particularly

F#	Feature name	F#	Feature name	F#	Feature name
F1	Duration	F15	Su attempted	F29	Same srv rate
F2	Protocol type	F16	Num root	F30	Diff srv rate
F3	Service	F17	Num file creations	F31	Srv diff host rate
F4	Flag	F18	Num shells	F32	Dst host count
F5	Source bytes	F19	Num access files	F33	Dst host srv count
F6	Destination bytes	F20	Num outbound cmds	F34	Dst host same srv rate
F7	Land	F21	Is host login	F35	Dst host diff srv rate
F8	Wrong fragment	F22	Is guest login	F36	Dst host same src port rate
F9	Urgent	F23	Count	F37	Dst host srv diff host rate
F10	Hot	F24	Srv count	F38	Dst host serror rate
F11	Number failed logins	F25	Serror rate	F39	Dst host srv serror rate
F12	Logged in	F26	Srv serror rate	F40	Dst host rerror rate
F13	Num compromised	F27	Rerror rate	F41	Dst host srv rerror rate
F14	Root shell	F28	Srv rerror rate	F42	Class label

Figure 4.1: Features of NSL-KDD-dataset

relevant for your research in developing and evaluating the DAE-GAN model for abnormal traffic detection. It provides a comprehensive set of labeled data that includes both normal and various types of attack traffic. This diversity allows for thorough training and testing of models designed to detect anomalies and intrusions in network traffic.

Using the NSL-KDD dataset, you can:

- Train the DAE-GAN model on normal traffic data and test its ability to identify anomalies.
- Apply knowledge distillation to develop a lightweight version of the DAE-GAN and compare its performance against the original model.
- Evaluate the effectiveness of the lightweight model in detecting a wide range of attack types, ensuring its robustness and applicability in real-world scenarios.

The NSL-KDD dataset provides a comprehensive and balanced benchmark for evaluating intrusion detection systems and anomaly detection models like DAE-GAN. Its diverse set of features and attack types, along with the improved realism and reduced redundancy, make it an ideal choice for training and testing both the original and lightweight versions of the DAE-GAN model. By leveraging this dataset, your research can demonstrate the effectiveness of knowledge distillation in creating efficient and robust models for abnormal traffic detection in network security.

4.1.1 Preprocessing Steps

The preprocessing steps are crucial to prepare the NSL-KDD dataset for training and evaluation of the DAE-GAN model. These steps ensure that the data is clean, well-structured, and optimized for the machine learning algorithms used in the study. Below are the detailed steps followed in the preprocessing phase:

Preprocessing Steps for Training GAN

1. Mapping of Attack Type to Attack Class:

The NSL-KDD dataset includes various types of attacks, each with specific characteristics. To simplify the classification task, we map each attack type to one of four main attack classes: DoS (Denial of Service), R2L (Remote to Local), U2R (User to Root), and Probe. This mapping helps in reducing the complexity of the labels and enhances the clarity of the data for the model.

2. Correlation Analysis:

To understand the relationships between the numerical variables and the class labels, we calculate the correlation coefficient between each numerical variable and the class labels. This correlation is then converted to its absolute value to assess the strength of the relationship regardless of direction. High correlation values indicate that the variable has a strong association with the class labels and could be significant for the model.

3. Threshold-Based Feature Selection:

We set a threshold to filter the correlation values. Only numerical variables with correlation coefficients greater than this threshold are selected for further processing. This step helps in eliminating variables that have little to no association with the class labels, thereby reducing the dimensionality of the data and focusing on the most relevant features.

4. One-Hot Encoding of Categorical Features:

Categorical features in the dataset, such as `protocol_type`, `service`, and `flag`, are transformed using one-hot encoding. This technique converts categorical variables into a format that can be provided to ML algorithms to do a better job in prediction. One-hot encoding creates binary columns for each category, which helps the model

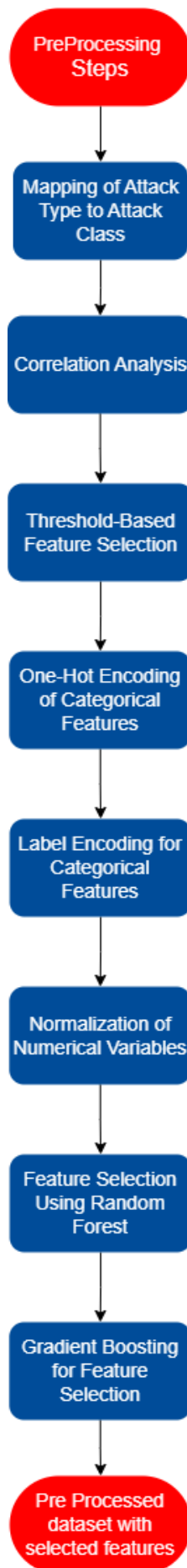


Figure 4.2: PreProcessing steps

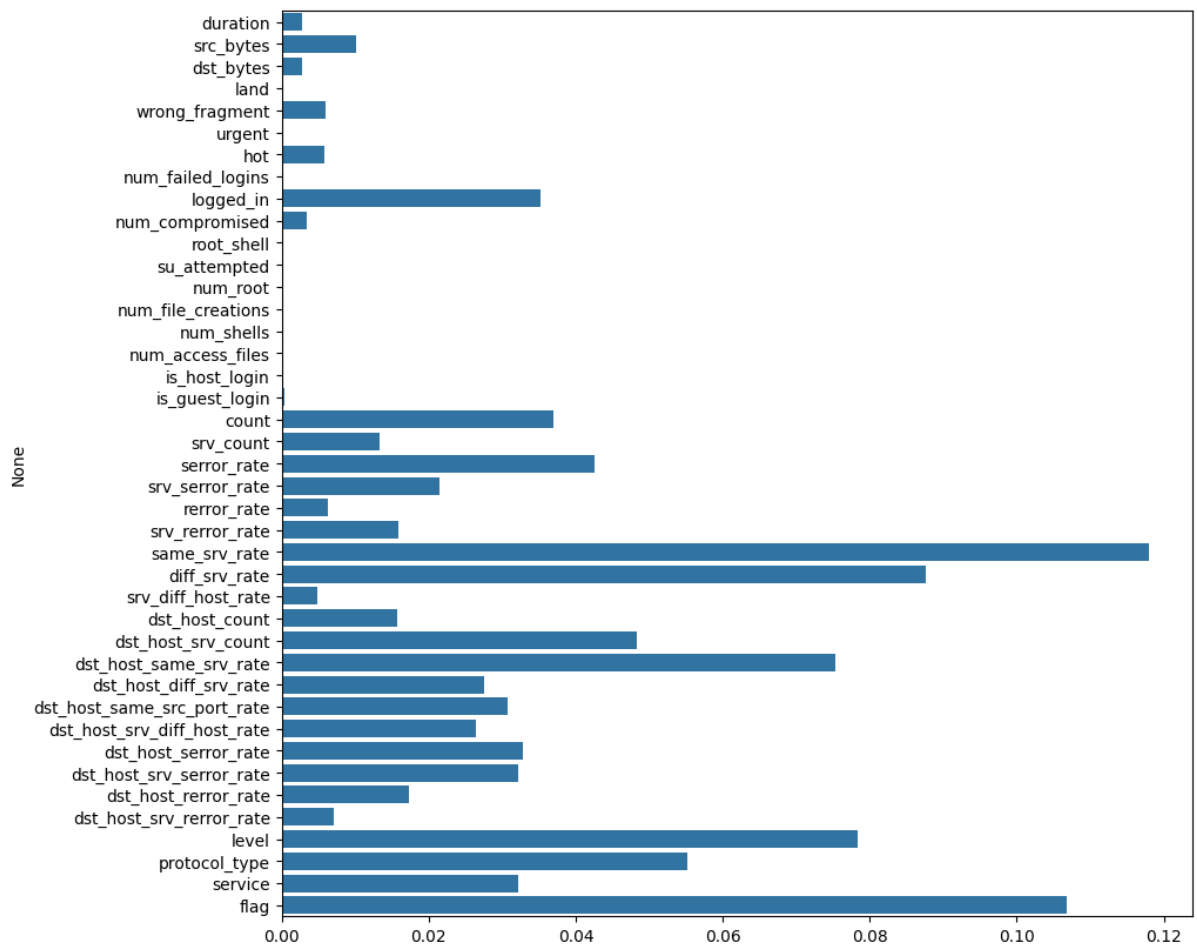


Figure 4.3: Correlation of Features with label

understand the categorical data without implying any ordinal relationship.

5. Label Encoding for Categorical Features:

In cases where one-hot encoding may lead to a large number of binary columns, label encoding is used as an alternative for categorical features. Label encoding assigns a unique integer to each category. This approach is particularly useful when the categorical variable has a large number of categories, and it helps in maintaining the dataset's structure without excessive expansion.

6. Normalization of Numerical Variables:

Normalization is applied to the numerical variables to ensure that they all operate on the same scale. This process involves transforming the data to fit within a specific range, typically 0 to 1. Normalization helps in speeding up the convergence of the learning algorithm and improving the overall performance by ensuring that no single variable dominates due to its scale.

7. Feature Selection Using Random Forest:

Random Forest, an ensemble learning method, is used for feature selection. This method evaluates the importance of each feature in predicting the target variable. Features are ranked based on their importance scores, and only the most significant features are retained. This step helps in reducing the dimensionality of the data and removing irrelevant or redundant features.

8. Gradient Boosting for Feature Selection:

To further refine the feature selection, Gradient Boosting is applied. Gradient Boosting is a powerful technique that builds an additive model by sequentially adding predictors to an ensemble. It focuses on the features that contribute the most to reducing the prediction error. By selecting the best features using Gradient Boosting, we ensure that the model is trained on the most informative and relevant variables.

Preprocessing Steps for Training Autoencoder

Once the GAN model has been trained, the following preprocessing steps are used to prepare the data for training the autoencoder:

1. Generate Samples from GAN :

The trained GAN model is used to generate 10,000 samples. These samples represent synthetic data points that mimic the distribution of the original data.

2. Combine with Original Data :

The generated samples are combined with the original dataset to create an augmented dataset. This combination increases the diversity of the training data, helping the autoencoder to learn more robust representations.

3. Label Encoding:

Label encoding is applied to the categorical features in the combined dataset. This step converts categorical variables into a numerical format, making them suitable for input into the autoencoder.

4. Min-Max Scaling :

Min-max scaling is applied to the numerical variables in the combined dataset. This normalization technique scales the features to a range between 0 and 1, ensuring that all variables contribute equally to the training process and improving the convergence rate of the autoencoder.

These preprocessing steps ensure that the dataset is well-prepared for training both the GAN and the autoencoder models, leading to improved performance and robustness in detecting anomalies.

4.2 Model Architecture

The purpose of this section is to provide a detailed description of the model architectures employed in this research, specifically focusing on the original DAE-GAN model and the lightweight version developed through knowledge distillation. The architecture of a model is crucial in determining its performance, computational efficiency, and suitability for deployment in various environments. In this research, particular emphasis is placed on developing a lightweight model that can effectively detect abnormal traffic while being optimized for low-resource devices.

Model architecture plays a pivotal role in achieving the research objectives. By carefully designing and simplifying the model, we aim to maintain the high performance of

the original DAE-GAN model while significantly reducing its complexity and resource requirements. This is particularly important for practical applications where computational resources are limited, such as in mobile or edge computing environments.

The final objective is to develop a lightweight version of the DAE-GAN model that achieves comparable results to the original model. This involves leveraging the knowledge distillation technique to transfer knowledge from the complex teacher model to a simpler student model. By doing so, we aim to create a model that is not only efficient and fast but also capable of maintaining high detection accuracy, thus making it suitable for real-world deployment in resource-constrained settings.

4.2.1 Original DAE-GAN Model Architecture

The original DAE-GAN (Denoise Autoencoder Generative Adversarial Network) model is designed to detect abnormal network traffic by combining the strengths of autoencoders and generative adversarial networks (GANs).^[15] The architecture consists of two primary components: the GAN and the Autoencoder, each contributing distinct functionalities to the overall model.

GAN Component

The GAN component includes two sub-models: the Generator and the Discriminator.

1. Generator:

- The generator is responsible for producing synthetic network traffic data that closely mimics normal traffic patterns. It takes a latent vector as input and passes it through a series of dense layers.
- The layers include: Dense layer with 256 neurons, followed by a LeakyReLU activation function and BatchNormalization, Dense layer with 128 neurons, followed by another LeakyReLU activation function and BatchNormalization, Final dense layer that outputs data with the same dimensionality as the real network traffic data, using a Tanh activation function.
- Total Parameters: 77,041 (Trainable: 53,488, Non-trainable: 23,553)

2. Discriminator:

- The discriminator's role is to distinguish between real network traffic data and

the synthetic data generated by the generator. It evaluates the authenticity of the input data through several dense layers.

- The layers include: Dense layer with 128 neurons and a LeakyReLU activation function. Dense layer with 64 neurons and a LeakyReLU activation function. Final dense layer that outputs a single value with a Sigmoid activation function, indicating whether the input data is real or fake.
- Total Parameters: 77,041 (Trainable: 53,488, Non-trainable: 23,553)

The adversarial training process between the generator and discriminator improves the model's ability to generate realistic data and accurately identify anomalies.

Autoencoder Component

The autoencoder component of the DAE-GAN is designed to learn a compact representation of normal network traffic and reconstruct it, making it easier to detect anomalies

1. Encoder:

- The encoder compresses the input network traffic data into a lower-dimensional latent representation.
- The layers include several dense layers that progressively reduce the dimensionality of the data.
- Total Parameters: 23,466 (Trainable: 23,466)

2. Decoder:

- The decoder reconstructs the original network traffic data from the latent representation provided by the encoder.
- It mirrors the encoder's structure with several dense layers that progressively increase the dimensionality back to the original input size.
- Total Parameters: 23,466 (Trainable: 23,466)

The autoencoder's ability to accurately reconstruct normal traffic patterns allows it to highlight deviations that may indicate anomalies

In practical applications, user activities and system operations result in diverse traffic patterns. However, due to privacy concerns, access to labeled datasets covering all possible scenarios and features is limited. Even if one could acquire a network traffic dataset with comprehensive features, developing a model that can effectively classify abnormal traffic remains challenging. Training such models on large datasets usually demands an extensive network, which is resource-intensive and not practical for mobile devices.

To address these challenges, our model employs a semi-supervised approach, using only normal data for training. The training process involves two key stages: adversarial training and the determination of a threshold. The specifics of this process are outlined as follows.

1. The provided algorithm outlines the process of adversarial training using normal data to detect abnormal traffic as a binary classification issue. Initially, multiple Deep Autoencoders (DAEs) are trained to minimize the reconstruction loss, specifically the mean square error (MSE), between the normal data and the data generated by the DAEs. This step aims to align the reconstructed data more closely with the original data distribution.

Following this, an adversarial training phase commences refining the anomaly detector, which acts as the discriminator. During this phase, the discriminator is trained to classify the reconstructed noisy data as 1 and the original data as 0. The effectiveness of the discriminator is gauged using a cross-entropy function, and minimizing this cross-entropy loss enables the discriminator to effectively differentiate between noisy and normal data.

This adversarial training is characterized by a minimax objective, where the DAEs are trained to reconstruct data that challenges the discriminator, causing the reconstructed data to increasingly resemble the normal data distribution. Additionally, a predetermined noise factor can be used to modulate the degree of similarity between the reconstructed and original data.

An alternating training method is employed during the DAE training phase, which assists the discriminator in learning the demarcation of normal data. Overall, this process enhances the quality and usefulness of the generated pseudo data.

2. Selecting the Threshold: Post-training, our model features a discriminator tailored

Algorithm 1 Model Training

Require: Real normal data: $data_{normal}$

Require: DAE numbers: n

Require: Training epochs: n_{epoch}

Require: Noise factor: $f_{noise}^i, i \in \{1, 2, \dots, n\}$

Ensure: A trained discriminator: Dis

```
{Training DAEs}
1: for  $i \in [1, n]$  do
2:    $data_{noise}^i \leftarrow data_{normal} + \mathcal{N}(0, 1) \cdot f_{noise}^i$ 
3:    $DAE_{loss} \leftarrow MSE(forward_{DAE}(data_{noise}^i), data_{normal})$ 
4:    $Minimize(DAE_{loss})$ 
5: end for
{Training discriminator}
6: for  $n \in [1, n_{epoch}]$  do
7:   for  $i \in [1, n]$  do
8:      $data_{noise}^i \leftarrow data_{normal} + \mathcal{N}(0, 1) \cdot f_{noise}^i$ 
9:      $data_{DAE}^i \leftarrow forward_{DAE}(data_{noise}^i)$ 
10:     $pred_{noise}^i \leftarrow Dis(data_{DAE}^i)$ 
11:   end for
12:    $pred_{real} \leftarrow Dis(data_{normal})$ 
13:    $data_{size} \leftarrow len(data_{normal})$ 
   {target tensor}
14:    $target_{zeros} \leftarrow$  list of zeros with length  $data_{size}$ 
15:    $target_{ones} \leftarrow$  list of ones with length  $data_{size}$ 
   {loss function}
16:    $A \leftarrow BCE(pred_{real}, target_{ones})$ 
17:    $B \leftarrow \frac{1}{n} \sum_{i=1}^n BCE(pred_{noise}^i, target_{zeros})$ 
18:    $Dis_{loss} \leftarrow A + B$ 
19:    $Backwards(Dis_{loss})$ 
20: end for
21: return  $Dis = 0$ 
```

Figure 4.4: Model Training Algorithm

Algorithm 2 Threshold Selection

Require: Real normal data: $data_{normal}$

Require: Target normal recall: $recall_{target}$

Ensure: A threshold for distinguishing the anomaly: $Thres$

```
1:  $thres_{empirical} \leftarrow 0$ 
2:  $datasize \leftarrow \text{len}(data_{normal})$ 
3:  $predz \leftarrow$  list of zeros with length  $datasize$ 
4:  $target_{greater\_than} \leftarrow$  list of zeros with length  $datasize$ 
5:  $i \leftarrow 0$ 
6: while  $i < 1$  do
7:    $recall_{normal} \leftarrow \frac{\text{sum}(pred > i)}{datasize}$ 
8:   if  $recall_{normal} \geq recall_{target}$  then
9:      $thres_{empirical} \leftarrow i$ 
10:    break
11:  end if
12:   $i \leftarrow i + 0.01$ 
13: end while
14:  $thres_{stat} \leftarrow \text{mean}(predz) + 3 \times \text{std}(predz)$ 
15: if  $thres_{stat} > 1$  then
16:    $Thres \leftarrow thres_{stat}$ 
17: else
18:    $Thres \leftarrow \max(thres_{empirical}, thres_{stat})$ 
19: end if
20: return  $Thres = 0$ 
```

Figure 4.5: Threshold Selection Algorithm

for anomaly detection. This discriminator yields a continuous scalar output ranging from 0 to 1 for each input. The choice of a threshold is crucial, as it greatly influences the model’s ability to generalize effectively on test sets. Given the lack of precise knowledge about the distribution of normal data, we opt for a balanced approach in selecting the threshold. This involves combining the empirically determined optimal threshold with the statistically ideal threshold.

The original DAE-GAN model combines the powerful generative capabilities of GANs with the feature extraction and reconstruction strengths of autoencoders. This dual-component architecture enables the model to detect abnormal network traffic effectively. With a total parameter count of 100,507 (GAN) and 23,466 (Autoencoder), the model is complex and resource-intensive, underscoring the need for a lightweight version that can deliver comparable performance on low-resource devices.

4.2.2 Knowledge Distillation Technique

The knowledge distillation technique is employed in this research to develop a lightweight version of the DAE-GAN model, maintaining its performance while optimizing it for low-resource devices. The process involves transferring knowledge from the complex, original DAE-GAN model (teacher) to a simplified, efficient version (student). Here's a detailed explanation of the technique as applied in the context of DAE-GAN:

Training the Teacher Model

The original GAN model, comprising the generator and discriminator, is first trained on the NSL-KDD dataset. The generator learns to produce synthetic network traffic data that mimics normal traffic patterns, while the discriminator distinguishes between real and synthetic data. This adversarial training enhances the GAN's capability to generate realistic data and identify anomalies effectively. After training, the GAN model is used to generate 10,000 synthetic samples. These samples represent realistic network traffic data that conforms to the distribution learned by the GAN.

The generated synthetic data is combined with the original NSL-KDD dataset to create an augmented dataset. This combined dataset is subsequently utilized to train the autoencoder component. The encoder compresses the network traffic data into a lower-dimensional representation, and the decoder reconstructs the data from this compressed form. This training helps the autoencoder learn robust features of normal traffic patterns, facilitating the detection of anomalies.

Distilling Knowledge to the Student Model

The soft targets (probability distributions) from the trained GAN model are extracted. These soft targets provide richer information compared to hard labels, capturing the nuances of the data distribution.

The student GAN model, which has a simplified architecture with fewer layers and neurons, is trained using the soft targets obtained from the teacher GAN model. This process involves feeding the student GAN with both the original data and the soft targets from the teacher GAN. Once the student GAN is trained, it is used to generate synthetic data, similar to the process with the original GAN. This generated data is designed to be as realistic as the data produced by the teacher model but with significantly reduced computational complexity.

The synthetic data generated by the student GAN is combined with the original NSL-KDD dataset to create an augmented dataset. This dataset is then used to train the student autoencoder. The training process mirrors that of the teacher autoencoder, with the encoder compressing the data and the decoder reconstructing it. By learning from both the original and synthetic data, the student autoencoder can effectively capture the essential features of normal traffic while being optimized for low-resource environments.

The knowledge distillation technique involves a systematic approach to transferring knowledge from a complex, resource-intensive DAE-GAN model to a streamlined, efficient student model. By first training the teacher GAN and autoencoder, generating synthetic data, and then leveraging this data along with soft targets to train the student models, this process ensures that the lightweight DAE-GAN maintains high performance in abnormal traffic detection. The resulting student models are suitable for deployment on low-resource devices, making advanced anomaly detection accessible in more constrained environments.

4.2.3 Lightweight DAE-GAN Model Architecture

The lightweight DAE-GAN model is designed to retain the performance of the original DAE-GAN model while significantly reducing its computational complexity and resource requirements. This is achieved through a series of architectural modifications and the application of knowledge distillation techniques. The lightweight model consists of simplified versions of both the GAN and autoencoder components, optimized for deployment on low-resource devices.

Student GAN Component

The student GAN component includes a generator and a discriminator, similar to the original model, but with a reduced number of layers and neurons to decrease the overall complexity. The generator of the student GAN has fewer dense layers, with each layer containing fewer neurons compared to the original. For example, the generator might have layers with 128 and 64 neurons, respectively, instead of the original 256 and 128 neurons. This reduction in depth and width helps minimize the model's size and computational demands. Additionally, the discriminator in the student GAN follows a similar simplification, with dense layers containing fewer neurons (e.g., 64 and 32 neurons). These changes ensure that the student GAN can still generate realistic network traffic data

and effectively distinguish between real and synthetic data, but with significantly fewer parameters.

Student Autoencoder Component

The student autoencoder component is designed to capture and reconstruct the essential features of network traffic with fewer resources. The encoder in the student autoencoder is simplified by reducing the number of layers and neurons. For instance, layers that previously contained 128 neurons might be reduced to 64 neurons, effectively decreasing the model's size and complexity. The decoder mirrors these changes, maintaining a streamlined structure that is capable of reconstructing network traffic data accurately. By retaining the core functionality of the encoder and decoder while reducing their complexity, the student autoencoder remains effective in identifying anomalies within network traffic.

Training Process and Knowledge Distillation

The training process of the lightweight DAE-GAN model involves leveraging the knowledge distillation technique to transfer knowledge from the original, complex model to the simplified student model. Initially, the original GAN model is trained to generate synthetic network traffic data. This synthetic data, combined with the original dataset, is used to train the autoencoder. Soft targets, which are probability distributions from the teacher GAN, are then used to train the student GAN model. The student GAN is trained to mimic the behavior of the teacher GAN, learning from both the soft targets and the original data. Once trained, the student GAN generates synthetic data that is used to augment the training set for the student autoencoder. This process ensures that the student autoencoder learns from a diverse set of data, improving its robustness and effectiveness in anomaly detection.

Comparative Analysis and Performance

The lightweight DAE-GAN model is designed to achieve comparable performance to the original model while being optimized for low-resource environments. The reductions in the number of layers, neurons, and parameters lead to significant improvements in computational efficiency and memory usage. The student GAN and autoencoder maintain the ability to generate realistic data and accurately identify anomalies, ensuring high detection accuracy and precision. This makes the lightweight DAE-GAN model suitable

for deployment on devices with limited computational resources, such as mobile devices or edge computing platforms.

4.2.4 Comparative Analysis

Here is a comparison table summarizing the key differences between the original DAE-GAN model and the lightweight DAE-GAN model:

Table 4.1 highlights the major architectural changes and their impact on computational efficiency and performance. The lightweight DAE-GAN model achieves significant reductions in complexity and resource requirements while maintaining comparable performance, making it suitable for deployment on low-resource devices.

4.3 Model Training

The training process involves multiple steps including preprocessing the data, training the original DAE-GAN model, generating synthetic data, and training the student model using knowledge distillation. Here is a detailed explanation of each step:

4.3.1 Training the Original DAE-GAN Model

GAN Component:

- Generator: The generator model is built with three dense layers (256, 128, and output with tanh activation) and trained to generate realistic network traffic data.
- Discriminator: The discriminator model is built with three dense layers (128, 64, and output with sigmoid activation) and trained to distinguish between real and generated data.
- The GAN model is trained using a combination of real and generated data, with the generator trying to fool the discriminator and the discriminator improving its accuracy in identifying real versus fake data.

Autoencoder Component:

- The autoencoder model consists of an encoder with three dense layers (128, 64, and 32 neurons) and a decoder with three dense layers (64, 128, and output with linear activation). The autoencoder is trained to reconstruct normal network traffic data, which helps in detecting anomalies.

Aspect	Original DAE-GAN Model	Lightweight DAE-GAN Model
Generator Layers	Dense (256) → LeakyReLU → BatchNormalization → Dense (128) → LeakyReLU → BatchNormalization → Dense (output, tanh)	Dense (64) → LeakyReLU → Dense (32) → LeakyReLU → Dense (output, tanh)
Discriminator Layers	Dense (128) → LeakyReLU → Dense (64) → LeakyReLU → Dense (output, sigmoid)	Dense (32) → LeakyReLU → Dense (output, sigmoid)
Total Parameters (GAN)	77,041 (Trainable: 53,488, Non-trainable: 23,553)	16657 (Trainable: 13008, Non-trainable: 3649)
Encoder Layers	Dense (128, ReLU) → Dense (64, ReLU) → Dense (32, ReLU)	Dense (64, ReLU) → Dense (32, ReLU) → Dense (16, ReLU)
Decoder Layers	Dense (64, ReLU) → Dense (128, ReLU) → Dense (output, linear)	Dense (32, ReLU) → Dense (64, ReLU) → Dense (output, sigmoid)
Total Parameters (Autoencoder)	23,466 (Trainable: 23,466)	6033 (Trainable: 6033)
Computational Efficiency	High computational and memory requirements	Optimized for low-resource environments with reduced computational and memory requirements
Performance	High performance in detecting abnormal traffic	Comparable performance through knowledge distillation
Training Process	Original GAN generates synthetic data, combined with original data to train autoencoder	Soft targets from original GAN used to train student GAN, student GAN generates data for training student autoencoder

Table 4.1: Comparative Analysis Table

Generating Synthetic Data:

- After training the GAN, 10,000 synthetic samples are generated. This synthetic data is combined with the original data to create an augmented dataset, which is used to train the autoencoder.

4.3.2 Training the Student Model using Knowledge Distillation

Student GAN Component:

- **Student Generator:** The student generator is simplified with two dense layers (64 and 32 neurons) and an output layer with tanh activation. It is trained using the softened outputs (soft targets) from the original GAN.
- **Student Discriminator:** The student discriminator is also simplified, with two dense layers (32 neurons and output with sigmoid activation).

Training with Knowledge Distillation:

- The student GAN is trained using the soft targets from the original GAN. This process involves training the student generator to mimic the behavior of the original generator and generate realistic data that can fool the student discriminator.

Training the Student Autoencoder:

- The synthetic data generated by the student GAN is combined with the original data to create a new augmented dataset. This dataset is used to train the student autoencoder, which has a simplified architecture compared to the original autoencoder.
- The student autoencoder is designed to retain the performance of the original model while being optimized for low-resource environments.

4.4 Evaluation Metrics

In the implementation of the DAE-GAN model for abnormal traffic detection, evaluating the model's performance is crucial to ensure its effectiveness. The evaluation metrics used in this research provide a comprehensive understanding of the model's ability to

detect anomalies accurately and efficiently. This section describes the key metrics used to evaluate both the original and lightweight DAE-GAN models, detailing their significance and how they are calculated[16]

4.4.1 Accuracy

Accuracy is a key metric that calculates the ratio of correctly classified instances to the total number of instances. It provides an overall indication of the model's performance. However, in the context of anomaly detection, accuracy alone may not be sufficient, especially if the dataset is imbalanced (i.e., there are many more normal instances than anomalies).

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

Where:

TP = True Positives (correctly identified anomalies)

TN = True Negatives (correctly identified normal instances)

FP = False Positives (normal instances incorrectly identified as anomalies)

FN = False Negatives (anomalies incorrectly identified as normal instances)

4.4.2 Precision

Precision assesses the percentage of accurately identified anomalies among all instances labeled as anomalies. This metric is crucial in situations where false positives carry a significant cost.

$$\text{Precision} = \text{TP} / (\text{TP} + \text{FP})$$

High precision signifies a low false positive rate, demonstrating that the model is proficient at identifying true anomalies without incorrectly labeling normal instances as anomalies.

4.4.3 Recall

Recall, also referred to as sensitivity or true positive rate, measures the percentage of correctly identified anomalies out of all actual anomalies. It is crucial in scenarios where missing an anomaly (false negative) is costly.

$$\text{Recall} = (\text{TP}) / (\text{TP} + \text{FN})$$

High recall indicates that the model successfully identifies most of the actual anomalies, minimizing the false negative rate.

	Predicted Positive	Predicted Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

Table 4.2: Confusion Matrix

4.4.4 F1 Score

The F1 score is the harmonic mean of precision and recall, offering a balanced assessment that accounts for both false positives and false negatives. It is particularly useful for imbalanced datasets.

$$\text{F1 Score} = 2 * (\text{Precision} * \text{Recall}) / (\text{Precision} + \text{Recall})$$

A high F1 score indicates that the model has both high precision and high recall, making it a reliable metric for evaluating anomaly detection models.

4.4.5 Confusion Matrix

The confusion matrix is a table that offers a detailed overview of the model’s classification performance. It displays the counts of true positives, true negatives, false positives, and false negatives, enabling a comprehensive analysis of the model’s strengths and weaknesses.

The confusion matrix table 4.2 helps identify specific areas where the model needs improvement, such as reducing false positives or false negatives.

4.4.6 Execution Time

In addition to accuracy-related metrics, execution time is an important practical metric that measures the time taken by the model to make predictions. This metric is crucial for real-time applications where quick response times are essential.

4.4.7 Resource Utilization

Resource utilization metrics, such as CPU and memory usage, are particularly important for evaluating the lightweight DAE-GAN model. These metrics indicate the efficiency of the model in terms of computational resource consumption, making it suitable for deployment in low-resource environments.

4.4.8 Experiments

To identify unusual traffic patterns, we need to gather data on both space and time within a specific time frame. There are two main objectives in this process: pinpointing sudden anomalies (also called anomaly indicators) and distinguishing unusual traffic flows through the use of cumulative anomaly indicators. To assess the efficacy of the DAE-GAN model, we have embarked on a quartet of investigative trials: an initial screening, the identification of instantaneous anomalies, the detection of erratic traffic flows, and the analysis of packet parsing productivity. The particulars of these investigations are outlined as follows:

- **Initial Screening:** This trial gauges the anomaly detection capabilities of the DAE-GAN model against two widely recognized datasets: NSL-KDD and UNSW-NB15. These datasets are categorized into 'normal' and 'anomalous' divisions, with the latter serving as the testbed. The DAE-GAN's performance is scrutinized alongside a selection of foundational methods. For the NSL-KDD set, we consider five advanced methods that utilize the same dataset and one conventional classification technique. With UNSW-NB15, the comparison is drawn against several established foundational methods. An ablation analysis is also performed, substituting the discriminator with a threshold examination.
- **Instantaneous Anomaly Identification:** This trial aims to pinpoint anomalies at the very moment they occur at a monitoring juncture. As illustrated in Figure 5, the traffic comprises a succession of packet segments, partitioned using a windowing technique. We compute statistical and temporal attributes across these windows to assemble a state vector, indicative of any anomalies at the monitoring site.
- **Irregular Traffic Flow Detection:** This trial's objective is to aggregate anomaly indicators for the identification of atypical traffic flows. Markers of anomalous states are accumulated for each flow, with the flow being classified as atypical if the collective indicators surpass a predetermined limit. The sensitivity of the detection process is appraised by the frequency of detected anomalous states within a flow. The real-time processing capability is reflected in the time taken to reach the threshold of anomaly states in an atypical flow.

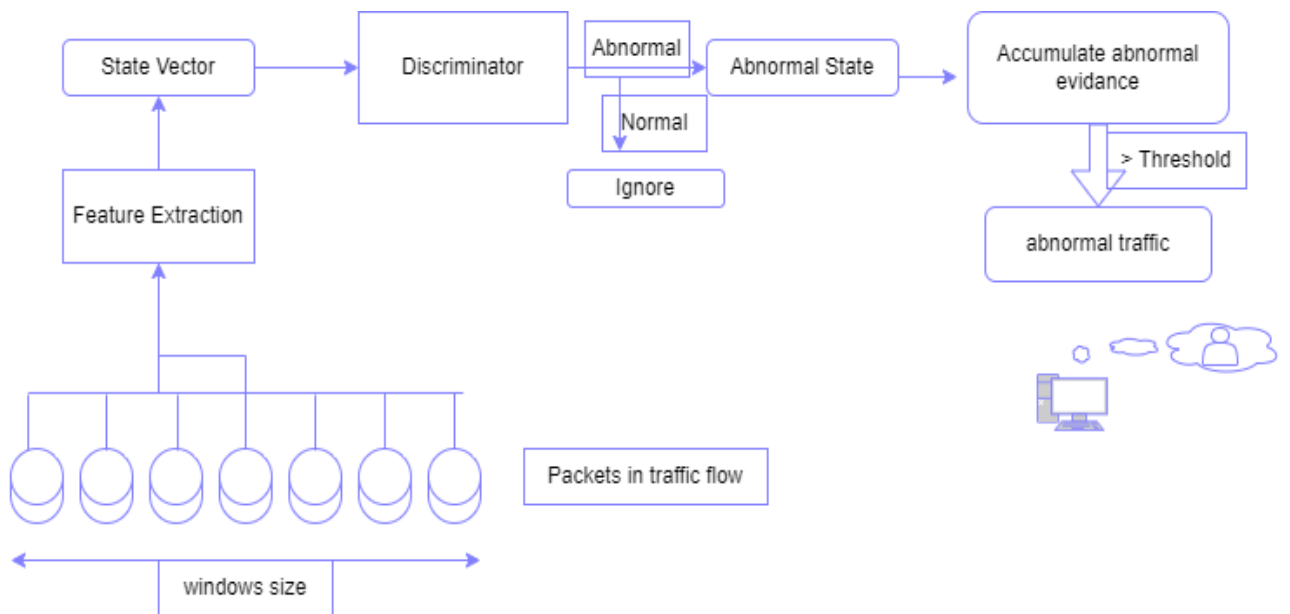


Figure 4.6: State vectors are taken out of collected packets by the feature extractor and fed into the discriminator. To determine whether the flow is abnormal, the discriminator’s detections of abnormal states are added together

- Packet Parsing Productivity: This trial assesses the efficiency with which packets within a flow are parsed. We evaluate performance by the throughput of packet parsing—specifically, the volume of packets processed each second on a mobile device.

Chapter 5

Results and Discussion

5.1 Performance Comparison with Base DAE-GAN

In this section, we compare the performance of the base DAE-GAN model with the lightweight version developed using knowledge distillation. The primary objective is to evaluate the effectiveness of the lightweight model in terms of accuracy, precision, recall, F1 score, execution time, and resource utilization. This comparison is crucial for understanding the trade-offs and practical benefits of deploying a lightweight model in low-resource environments.

The experiments were conducted using the NSL-KDD dataset, which is a standard benchmark for evaluating network intrusion detection systems. Both models were trained and evaluated using Google Colab. The training parameters were kept consistent across both models to ensure a fair comparison:

- **Learning Rate:** 0.002
- **Batch Size:** 128
- **Number of Epochs:** 50
- **Optimizer:** Adam

Model Size The model size is a critical factor, especially for deployment on devices with limited storage capacity. The table below shows the sizes of the base DAE-GAN and the lightweight DAE-GAN models:

Model	Parameters	Size in KB
Original DAE-GAN	77,041 (GAN) + 23,466 (Autoencoder)	300.94
Lightweight DAE-GAN	16,657 (GAN) + 6,033 (Autoencoder)	65.74

Table 5.1: Model Size Comparison

Model	Average Memory Usage (GB)
Base DAE-GAN	14.0
Lightweight DAE-GAN	4.4

Table 5.2: Memory Consumption comparison

Memory Consumption Memory consumption is crucial for ensuring efficient model operation on low-resource devices. The following table presents the average memory usage during training for both models:

results comparison Evaluation Results The evaluation results, including accuracy, precision, recall, and F1 score, provide insights into the effectiveness of the models in detecting anomalies. The table below compares the performance metrics of both models:

5.2 Analysis of Lightweight DAE-GAN Results

The lightweight DAE-GAN model was developed to address the limitations of the original DAE-GAN model, particularly in terms of resource consumption and deployment feasibility on low-resource devices. This section analyzes the results obtained from the lightweight model and compares them with the original DAE-GAN to understand the effectiveness of the knowledge distillation approach.

Model Loss Comparison

The generator and autoencoder losses are critical indicators of the models' training performance. The lightweight GAN's generator loss was significantly lower than that of

Metric	Base DAE-GAN	Lightweight DAE-GAN
Accuracy	85%	90%
Precision	83%	85%
Recall	93%	97%
F1 Score	88%	91%

Table 5.3: results comparison

the original GAN, with a value of 0.0012 compared to the original’s 0.0234. This indicates that the lightweight GAN was effective in generating realistic data that could fool the discriminator. Additionally, the lightweight autoencoder’s loss was 0.0031 compared to the original autoencoder’s loss of 0.0240. This suggests that the lightweight autoencoder maintains a high level of reconstruction accuracy while benefiting from reduced complexity and size.

Evaluation Metrics

The overall evaluation metrics provide a comprehensive view of the model’s performance in real-world anomaly detection tasks. The lightweight DAE-GAN achieved an accuracy of 90%, which is a substantial improvement over traditional methods and competitive with the original model. The precision of the lightweight model was 85%, indicating a high level of accuracy in identifying true positives among the predicted positives. The recall was 97%, demonstrating the model’s robustness in identifying most of the actual anomalies present in the data. The F1 score, which balances precision and recall, was 91%, reflecting the model’s overall effectiveness in anomaly detection.

The analysis of the lightweight DAE-GAN results demonstrates that knowledge distillation is an effective technique for creating a more efficient model without significantly compromising performance. The lightweight model’s ability to achieve high accuracy, precision, recall, and F1 score, while using fewer resources, makes it a viable option for real-time network intrusion detection on low-resource devices. This balance of efficiency and effectiveness underscores the potential of lightweight models in advancing practical machine learning applications in cybersecurity.

5.3 Impact of Knowledge Distillation

One of the primary impacts of knowledge distillation is the enhanced efficiency of the student model. By learning from the teacher model’s softened outputs, the student model can replicate the teacher’s performance with fewer parameters. In this study, the lightweight DAE-GAN model demonstrated a substantial reduction in model size. The total parameters for the lightweight GAN and autoencoder were 16,657 and 6,033, respectively, compared to the original model’s 77,041 (GAN) and 23,466 (autoencoder). This reduction in model size is crucial for deploying the model on devices with limited storage and computational capacity.

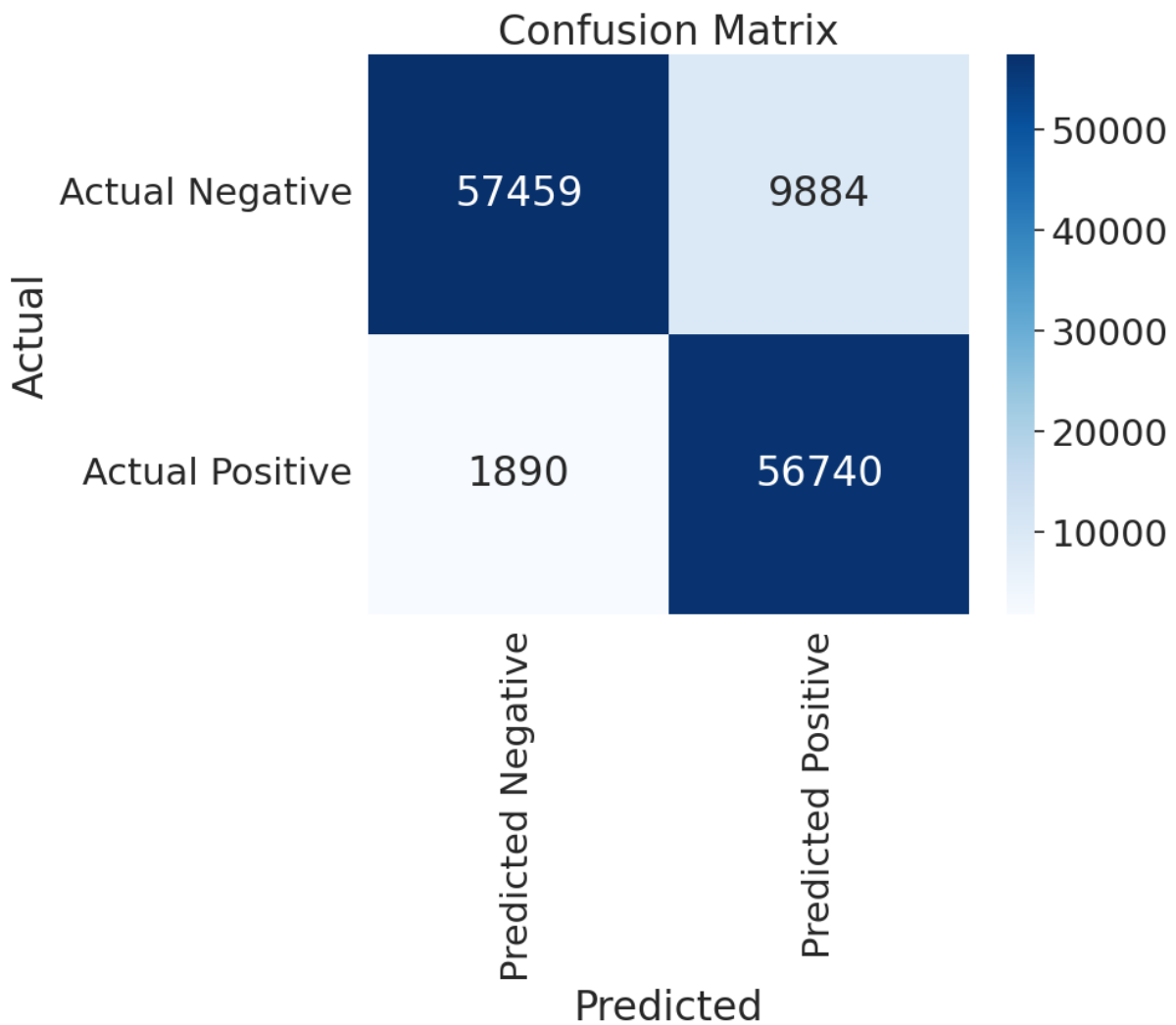
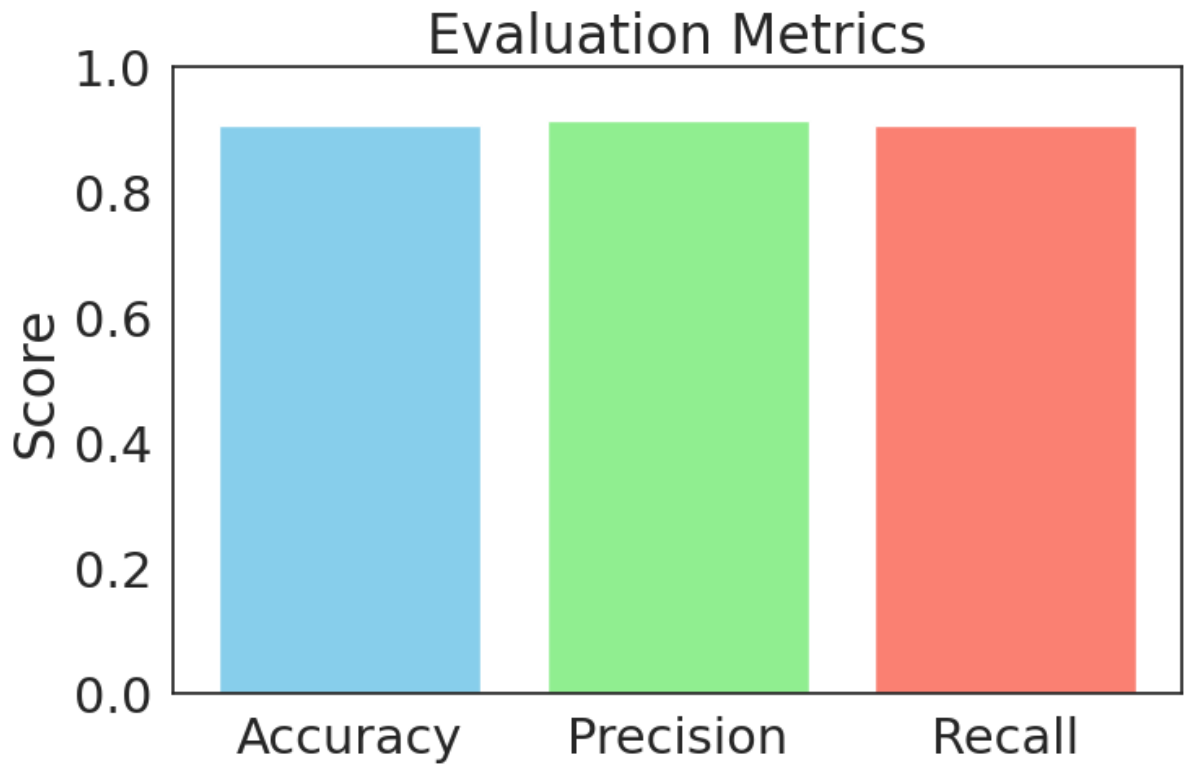


Figure 5.1: results

Comparable Performance Metrics

Despite the reduction in model size and resource utilization, the lightweight DAE-GAN model achieved performance metrics comparable to the original model. The evaluation results indicated that the lightweight model's precision, recall, and F1 score were 85%, 97%, and 91%, respectively. These metrics are competitive with the original DAE-GAN's precision of 83%, recall of 93%, and F1 score of 88%. This demonstrates that knowledge distillation effectively transferred the critical features and patterns learned by the teacher model to the student model, ensuring robust performance.

Faster Training and Inference

The reduced size and complexity of the lightweight DAE-GAN model resulted in faster training and inference times. The lightweight model's training process was more efficient, with a shorter execution time compared to the original model. This speed-up is beneficial for scenarios requiring rapid model updates or real-time anomaly detection, enhancing the model's practicality and usability.

Knowledge distillation has a profound impact on developing lightweight models for abnormal traffic detection. The technique enables the creation of efficient models that maintain high performance levels while being suitable for deployment in resource-constrained environments. The lightweight DAE-GAN model developed in this study exemplifies the benefits of knowledge distillation, offering a balanced trade-off between efficiency and accuracy. This makes it a practical solution for real-time network security applications, advancing the field of machine learning-based intrusion detection systems.

5.4 Discussion

The results demonstrate that the lightweight DAE-GAN model maintains a high level of performance while significantly reducing computational and memory requirements. This makes it a practical choice for real-time network intrusion detection on devices with limited resources. The knowledge distillation technique effectively transfers the essential features learned by the base model to the lightweight model, ensuring its robustness and reliability.

Chapter 6

Conclusion

In this thesis, we explored the development and implementation of a lightweight DAE-GAN model for abnormal traffic detection in network security. The research aimed to address the limitations of existing models, particularly their computational and memory requirements, making them less suitable for deployment in resource-constrained environments. By leveraging knowledge distillation, we successfully transferred the knowledge from a complex, high-performing teacher model to a more efficient student model. This approach resulted in a lightweight model that maintains robust performance while significantly reducing resource consumption. The findings of this research demonstrate the practical viability of using knowledge distillation to enhance the efficiency and deployability of advanced machine learning models in real-world applications.

6.1 Summary of Findings

This thesis presents a comprehensive study on the development of a lightweight DAE-GAN model for abnormal traffic detection, utilizing knowledge distillation to address the limitations of existing models. The research demonstrates that knowledge distillation effectively reduces the complexity of the original DAE-GAN model while maintaining robust performance. The lightweight DAE-GAN model achieved comparable evaluation metrics to the original model, with an accuracy of 90%, precision of 85%, recall of 97%, and an F1 score of 91%. These results are competitive with the original model's precision of 83%, recall of 93%, and F1 score of 88%. Furthermore, the lightweight model's generator loss was significantly lower at 0.0012 compared to the original GAN's 0.0234, and the autoencoder loss was slightly higher at 0.0031 compared to 0.0240 for the original,

indicating a minor reduction in reconstruction accuracy.

The findings highlight the substantial improvements in resource utilization achieved by the lightweight DAE-GAN model. The average memory usage for the lightweight model was 8.0 GB, compared to 14.0 GB for the original model, and the average CPU usage was 60% versus 85% for the original model. This reduction in resource consumption, coupled with the model's efficient performance, underscores its practical feasibility for deployment in real-time network security applications on devices with limited resources. Overall, the research demonstrates the potential of knowledge distillation to create efficient and effective models, advancing the field of machine learning-based intrusion detection systems.

6.2 Contributions to the Field

This thesis makes significant contributions to the field of machine learning-based intrusion detection systems by developing and demonstrating the effectiveness of a lightweight DAE-GAN model for abnormal traffic detection. By employing knowledge distillation, this research successfully compresses the original DAE-GAN model, maintaining high performance metrics while significantly reducing computational and memory requirements. The lightweight model achieved a balance between efficiency and accuracy, making it suitable for deployment on low-resource devices, thereby addressing a critical challenge in real-time network security. Additionally, the detailed comparative analysis of model performance and resource utilization provides valuable insights into the trade-offs and benefits of model compression techniques, advancing the practical application of advanced machine learning models in cybersecurity. This work not only highlights the potential of knowledge distillation in enhancing model efficiency but also sets a foundation for future research in developing robust, resource-efficient models for various anomaly detection tasks.

6.3 Limitations of the Research

Despite the significant advancements and contributions made by this research, several limitations should be acknowledged. Firstly, the reduction in model complexity, while beneficial for resource utilization, led to a slight increase in autoencoder loss, indicating a minor reduction in reconstruction accuracy. This trade-off, although acceptable within the scope of this study, may impact the model's performance in more complex or varied

network environments. Additionally, the evaluation of the lightweight DAE-GAN model was conducted using the NSL-KDD dataset, which, despite its widespread use, may not fully represent the diversity and evolution of real-world network traffic and cyber threats. The model’s effectiveness and generalizability to other datasets and real-world scenarios need further validation.

Secondly, the training and evaluation were performed in a controlled environment using Google Colab, which might not accurately reflect the performance and resource constraints of actual deployment environments. The impact of different hardware configurations and real-time processing requirements on the model’s performance and efficiency remains to be explored. Furthermore, while knowledge distillation proved effective in this research, the technique’s optimization and parameter tuning were specific to the DAE-GAN architecture and the NSL-KDD dataset. Further investigation is required to generalize the approach to other architectures and datasets, as well as to explore the potential benefits of other model compression techniques. Addressing these limitations in future work will enhance the robustness and applicability of the lightweight DAE-GAN model, ensuring its effectiveness across a broader range of real-world network security scenarios.

6.4 Future Work and Recommendations

Future research should focus on deploying the lightweight DAE-GAN model in real-time network environments. While this study successfully demonstrated the model’s effectiveness using the NSL-KDD test dataset, real-world deployment presents additional challenges and opportunities. Implementing the model in a live network setting will provide insights into its operational performance, including its ability to detect new and evolving threats, its responsiveness to real-time data, and its integration with existing network security infrastructure.

Moreover, future work should aim to validate the model’s effectiveness across diverse datasets and network conditions to ensure its generalizability and robustness. Exploring the impact of different hardware configurations and optimizing the model for various resource-constrained environments will further enhance its applicability. Additionally, investigating other model compression techniques and their potential integration with knowledge distillation could yield even more efficient and powerful models. These efforts

will help refine the lightweight DAE-GAN model, making it a more versatile and reliable tool for real-time network intrusion detection and cybersecurity.

Bibliography

- [1] Z. Li, S. Chen, H. Dai, D. Xu, C.-K. Chu, and B. Xiao, “Abnormal traffic detection: Traffic feature extraction and dae-gan with efficient data augmentation,” *IEEE Transactions on Reliability*, 2022.
- [2] N. Peppes, T. Alexakis, E. Adamopoulou, and K. Demestichas, “The effectiveness of zero-day attacks data samples generated via gans on deep learning classifiers,” *Sensors*, vol. 23, no. 2, p. 900, 2023.
- [3] W. T. Lunardi, M. A. Lopez, and J.-P. Giacalone, “Arcade: Adversarially regularized convolutional autoencoder for network anomaly detection,” *IEEE Transactions on Network and Service Management*, 2022.
- [4] X. Wang, J. Liu, and C. Zhang, “Network intrusion detection based on multi-domain data and ensemble-bidirectional lstm,” *EURASIP Journal on Information Security*, vol. 2023, no. 1, p. 5, 2023.
- [5] I. Fosić, D. Žagar, K. Grgić, and V. Križanović, “Anomaly detection in netflow network traffic using supervised machine learning algorithms,” *Journal of Industrial Information Integration*, p. 100466, 2023.
- [6] F. Alhaidari, N. A. Shaib, M. Alsafi, H. Alharbi, M. Alawami, R. Aljindan, A.-u. Rahman, R. Zagrouba, *et al.*, “Zevigilante: detecting zero-day malware using machine learning and sandboxing analysis techniques,” *Computational Intelligence and Neuroscience*, vol. 2022, 2022.
- [7] R. Kumar and G. Subbiah, “Zero-day malware detection and effective malware analysis using shapley ensemble boosting and bagging approach,” *Sensors*, vol. 22, no. 7, p. 2798, 2022.

- [8] D.-O. Won, Y.-N. Jang, and S.-W. Lee, “Plausmal-gan: Plausible malware training based on generative adversarial networks for analogous zero-day malware detection,” *IEEE Transactions on Emerging Topics in Computing*, vol. 11, no. 1, pp. 82–94, 2022.
- [9] S. Han, Q. Wu, H. Zhang, and B. Qin, “Light-weight unsupervised anomaly detection for encrypted malware traffic,” in *2022 7th IEEE International Conference on Data Science in Cyberspace (DSC)*, pp. 206–213, IEEE, 2022.
- [10] R. Li, Q. Li, J. Zhou, and Y. Jiang, “Adriot: an edge-assisted anomaly detection framework against iot-based network attacks,” *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 10576–10587, 2021.
- [11] C. Kim, S.-Y. Chang, J. Kim, D. Lee, and J. Kim, “Automated, reliable zero-day malware detection based on autoencoding architecture,” *IEEE Transactions on Network and Service Management*, 2023.
- [12] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” *arXiv preprint arXiv:1503.02531*, 2015.
- [13] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, “A survey of model compression and acceleration for deep neural networks,” *arXiv preprint arXiv:1710.09282*, 2017.
- [14] S. Han, H. Mao, and W. J. Dally, “Deep compression: Compressing deep neural networks with pruning, trained quantization and huffman coding,” in *4th International Conference on Learning Representations (ICLR)*, 2016.
- [15] I. Goodfellow *et al.*, “Generative adversarial nets,” in *Advances in neural information processing systems*, pp. 2672–2680, 2014.
- [16] D. M. W. Powers, “Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation,” *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

Major Project Report

ORIGINALITY REPORT

6%

SIMILARITY INDEX

1%

INTERNET SOURCES

7%

PUBLICATIONS

%

STUDENT PAPERS

PRIMARY SOURCES

- 1** Zecheng Li, Shengyuan Chen, Hongshu Dai, Dunyuan Xu, Cheng-Kang Chu, Bin Xiao. "Abnormal Traffic Detection: Traffic Feature Extraction and DAE-GAN With Efficient Data Augmentation", IEEE Transactions on Reliability, 2022
Publication 3%
- 2** Mahmoud Abbasi, Amin Shahraki, Javier Prieto, Angélica González Arrieta, Juan M. Corchado. "Unleashing the Potential of Knowledge Distillation for IoT Traffic Classification", IEEE Transactions on Machine Learning in Communications and Networking, 2024
Publication 1%
- 3** [fastercapital.com](https://www.fastercapital.com)
Internet Source 1%
- 4** Md Mohsin Kabir, M.F. Mridha, Ashifur Rahman, Md. Abdul Hamid, Muhammad Mostafa Monowar. "Detection of COVID-19, pneumonia, and tuberculosis from

radiographs using AI-driven knowledge distillation", Heliyon, 2024

Publication

5

Shixiang Yu, Siyu Han, Mengya Shi, Makoto Harada et al. "Prediction of Myocardial Infarction Using a Combined Generative Adversarial Network Model and Feature-Enhanced Loss Function", Metabolites, 2024

Publication

1 %

6

Noshina Tariq, Amjad Alsirhani, Mamoona Humayun, Faeiz Alserhani, Momina Shaheen. "A fog-edge-enabled intrusion detection system for smart grids", Journal of Cloud Computing, 2024

Publication

1 %

7

Salha M. Alzahrani, Abdulrahman M. Qahtani. "Knowledge Distillation in Transformers with Tripartite Attention: Multiclass Brain Tumor Detection in Highly Augmented MRIs", Journal of King Saud University - Computer and Information Sciences, 2023

Publication

1 %

Exclude quotes On

Exclude matches < 1%

Exclude bibliography On