الجمهورية الجزائرية الديمقراطية الشعبية

**Ministry Of Higher Education and Scientific Research**

**Yahia Fares University of Medea**

**Faculty of Sciences**

**Department of Mathematics and Computer Science**



Graduation Project Report Submitted for Master of Computer Science Degree

Specialty : Engineering Systems and Web Technology

---

# Intrusion Detection System Using DL for Iot Devices

---

***Proposed and directed by:***

Dr. Ahmed Cherif MAZARI

***Prepared and presented by:***

Ms . Rania Chirez Hamdane

Ms . Nour El Houda Ettouahri

# THANKS

At the end of this work, we wish to express our profound gratitude to our dear professor and advisor, Mr. Mazari Ahmed Cherif, for his attentive guidance, patience, and the valuable information he has continuously provided throughout the project period.

We also extend my sincere thanks to the members of the jury who have taken the time to evaluate this work.

# Dedicace

Je commence par remercier Dieu, le Tout-Puissant, pour Son aide et pour m'avoir donné la capacité de poursuivre et de chercher le savoir.

Avec gratitude et sincères remerciements, je dédie ce modeste travail

À ma chère maman, source de mon énergie, refuge face aux difficultés de la vie, raison de mon existence, et première école.

À mon cher papa, pour son encouragement, ses efforts et ses sacrifices.

Mes chers parents, je ne pourrai jamais vous remercier assez pour tout ce que vous avez fait pour moi et pour ce que vous avez fait de moi. Ce travail n'est qu'un petit fruit de vos efforts constants à mes côtés.

À mes chères sœurs, Nadine et Nahla qui partagent la joie de mon succès.

À Ma grande mère et ma tante adorée Mounira qui m'encouragent toujours .

Que Dieu vous garde pour moi.

À mes enseignants, mes collègues avec qui j'ai partagé de nombreuses années d'études,
et à tous ceux qui m'ont un jour apporté leurs aides

Que Dieu vous récompense pour tous vos bienfaits.

Enfin, à tous ceux que j'aime et qui m'aiment, je dédie ce travail.

Rania Chiraz Hamdane

# Dedicace

I dedicate this dissertation

To my parents, for their invaluable love, their trust, their support, their sacrifices and all the values they knew how to instill in me.

To my brothers, for their tenderness, their understanding and their presence.

To my ETTOUHARI family.

To my husband, who believes in the beauty of dreams and the power of perseverance, for his valuable encouragement and advice.

To all friends and colleagues in the specialty.

Ettouahri Houda

# Table of Contents


Tables List ..... 10

Figures List ..... 11

Acronyms List ..... 13

Abstract ..... 16

Introduction ..... 19

I Chapter I : Background and IoT security ..... 22

I.1 Introduction ..... 23

I.2 IoT Definition ..... 23

I.3 IoT Architecture ..... 23

I.4 Importance and Application Domains of the IoT ..... 25

I.4.1 Smart Home ..... 26

I.4.2 Smart City ..... 26

I.4.3 Industrial IoT ..... 26

I.4.4 Smart Grid ..... 26

I.4.5 Agriculture ..... 26

I.4.6 Health care IoT ..... 27

I.5 IT security ..... 27

I.6 Security Attacks in IoT ..... 27

I.7 How to protect computer systems ..... 30

I.8 IoT Intrusions ..... 31

II Chapter II: ..... 34

Intrusion detection systems Using DL ..... 34

II.1 Motivation ..... 35

II.2 Intrusion detection system ..... 35

II.3 Information source based IDS ..... 38

*II.3.1* Network Intrusion Detection System (NIDS) ..... 39

II.3.2 Host Intrusion Detection System (HIDS) ..... 39

II.3.3 Hybrid Intrusion Detection System (Hybrid IDS) ..... 39

II.4 Detection based IDS ..... 40

II.4.1 Signature based intrusion detection ..... 40

II.4.2 Anomaly based intrusion detection ..... 40

II.5 Security challenges in IoT and IoT IDS ..... 40

II.6 IDS and Deep Learning for IoT Security ..... 41

II.7 Comparison between deep learning and traditional machine learning ..... 43
II.8 Deep learning for IoT ..... 44
II.8.1 Deep learning classification ..... 45
II.8.2 Deep Learning models ..... 46
II.9 Cybersecurity Datasets ..... 51
II.10 Related Works ..... 53
III Chapter III ..... 64
Design of the Models Used ..... 64
III.1 Introduction ..... 65
III.2 Preprocessing and Data Analysis ..... 67
III.3 CICIDS2017 Dataset Description ..... 68
III.4 CICDDDS2019 Dataset Description ..... 68
III.5 CICFlowMeter ..... 70
III.6 Data Preprocessing ..... 70
III.7 Proposed method for columns dimensionality reduction ..... 72
III.8 Data classification: ..... 74
III.8.1 One-Dimensional Convolutional Neural Network (1D-CNN) ..... 75
III.8.2 Two-Dimensional Convolutional Neural Network (2D-CNN) ..... 75
III.9 The Hyperparameters used for CNN classifications ..... 77
III.10 Feed-Forward Deep Neural Network (FFDNN) ..... 79
III.11 Long Short-Term Memory (LSTM) ..... 80
III.12 Visual Geometry Group (VGG) ..... 82
III.13 MobileNet V2 ..... 83
III.14 DenseNet 121 ..... 84
III.15 Performance Indicators ..... 84
IV Chapter IV ..... 88
Experimental Results and Discussion ..... 88
IV.1 Introduction ..... 89
IV.2 Experimental Datasets, Deep Learning Models used and Evaluation Metrics ..... 89
IV.3 Data Preprocessing ..... 90
IV.4 Feature Extraction ..... 91
IV.4.1 Extracting Features Using Autoencoder ..... 91
IV.4.2 Extracting Features Using VGG16 ..... 94
IV.5 Classification using Deep Learning Models ..... 94
IV.5.1 AE+CNN 1D Model ..... 94
IV.5.2 AE+CNN 2D Model ..... 98

IV.5.3 AE+FFDNN Model ..... 101
IV.5.4 AE+LSTM Model ..... 103
IV.5.5 Combination of Vgg16 , MobileNetV2 and DenseNet121 Model ..... 105
IV.6 Discussion ..... 108
IV.6.1 Analyze of results obtained ..... 108
IV.6.2 Comparison with existing related works ..... 109
V CONCLUSION AND FUTURE WORKS ..... 112
VI Bibliography ..... 113

# Tables List

**Table 1:**Types of attacks and intrusions affecting each layer of the IoT architecture ..... 32
**Table 2**:Evolution and Classification of Intrusion Detection Systems (IDS) ..... 36
**Table 3:**Related works ..... 55
**Table 4:**Steps for the Intrusion Detection System Conception ..... 66
**Table 5:**Normalization and Balancing Steps ..... 72
**Table 6:**Steps to reshape tabular data into 4D tensor ..... 76
**Table 7:**Hyperparameters of the proposed DL models ..... 78
**Table 8:**Number of rows before and after balancing the dataset ..... 90
**Table 9:** Parameters and output shapes of the autoencoder model ..... 91
**Table 10:**Parameters and output shapes of the CNN 1D model ..... 95
**Table 11:**Results of AE+CNN 1D Model on the CicIDS 2017 dataset and CicDDos 2019 dataset ... 95
**Table 12:**Parameters and output shapes of the CNN 2D model ..... 98
**Table 13**:Results of AE+CNN 2D Model on the CicIDS 2017 dataset and CicDDos 2019 dataset ... 98
**Table 14:**Parameters and Output shapes of the FFDNN Model ..... 101
**Table 15:**Results of FFDNN Model on CicIDS2017 ..... 102
**Table 16**:Parameters of LSTM model created ..... 103
**Table 17:** Results of AE+LSTM Model on the CicIDS 2017 dataset ..... 104
**Table 18:** Parameters and values of the model component ..... 105
**Table 19:** Results of Vgg16 + MobileNetV2 + Densenet Model on the CicIDS 2017 dataset ..... 106
**Table 20**:Summury of the work results ..... 108

# Figures List

**Figure 1:Architecture of IoT (A: three layers) (B: five layers)** ..... 24
**Figure 2:Application Domains of loT** ..... 25
**Figure 3:An example IDS for DDoS attacks** ..... 28
**Figure 4:Types of IDS** ..... 38
**Figure 5:Classification of intrusion detection systems** ..... 38
**Figure 6:IDS using deep learning models** ..... 42
**Figure 7:Classification of deep learning architecture** ..... 45
**Figure 8:Existing deep learning models** ..... 46
**Figure 9:DNN, RNN, CNN, and LSTM Architectures** ..... 49
**Figure 10:** DBN, GAN, DRL architectures ..... 51
**Figure 11:Diagram of the shared and exclusive features of four IDS datasets** ..... 53
**Figure 12:Plan of creation IDS using Deep learning** ..... 65
**Figure 13:Statistics of attacks and normal behavior in the CICIDS 2017 dataset** ..... 68
**Figure 14:Description of CICIDS2017 Dataset** ..... 71
**Figure 15:Autoencoder architecture.** ..... 73
**Figure 16:Exemple of architecture of CNN 1D model** ..... 75
**Figure 17:Multi layer feed-forward network** ..... 80
**Figure 18: General LSTM architecture** ..... 81
**Figure 19:Vgg16 Architecture** ..... 82
**Figure 20:Architecture of MobilenetV2** ..... 83
**Figure 21:Confusion matrix** ..... 84
**Figure 22:Exemple of Training and validation accuracy Graph** ..... 86
**Figure 23:MSE for both training and validation sets over 17 epochs** ..... 93
**Figure 24:Confusion Matrix of AE+CNN 1D Model on CicIDS2017 Dataset** ..... 96
**Figure 25:Confusion Matrix of AE+CNN 1D Model on CicDDOS2019 Dataset** ..... 96
**Figure 26:Train and Validation Accuracy Graph of AE+CNN 1D Model** ..... 97
**Figure 27:Train and Validation Accuracy Graph of AE+CNN 1D Model on CicDDOS2019 Dataset** ..... 97
**Figure 28:Confusion Matrix of AE+CNN 2D Model on CicIDS2017 Dataset** ..... 99
**Figure 29:Confusion Matrix of AE+CNN 2D Model on CicDDOS2019 Dataset** ..... 99
**Figure 30:Train and Validation Accuracy Graph of AE+CNN 2D Model** ..... 100
**Figure 31:Train and Validation Accuracy Graph of AE+CNN 2D Model** ..... 100

**Figure 32:Confusion matrix of FFDNN Model on CicIDS2017 dataset** ..... 102
**Figure 33:Train and Validation Accuracy Graph of AE+FFDNN Model** ..... 103
**Figure 34:Confusion Matrix of AE+LSTM Model on CicIDS2017 Dataset** ..... 104
**Figure 35:Train and Validation Accuracy Graph of AE+LSTM Model** ..... 105
**Figure 36:Confusion matrix of Vgg16 , MobileNetV2 and DenseNet 121 Model on CicIDS2017 Dataset** ..... 107
**Figure 37:Train and Validation Accuracy Graph of Vgg16 , MobileNetV2and DenseNet 121 Model on CicIDS2017 Dataset** ..... 107

# Acronyms List

**IDS :** Intrusion Detection System

**DL :** Deep Learning

**IoT :** Internet of Things

**AE :** Autoencoder

**CNN :** Convolutional Neural Network

**LSTM :** Long Short-Term Memory

**FFDNN :** Feed-Forward Deep Neural Network

**SMOTE :** Synthetic Minority Over-sampling Technique

**CICIDS :** Canadian Institute for Cybersecurity Intrusion Detection System

**HIDS :** Host Intrusion Detection System

**NIDS :** Network Intrusion Detection System

**AI :** Artificial Intelligence

**ML :** Machine Learning

**RNN :** Recurrent Neural Network

**RBM :** Restricted Boltzmann Machine

**DBN :** Deep Belief Network

**GAN** : Generative Adversarial Network

**DRL** : Deep Reinforcement Learning

**PCAP :** Packet Capture

**IoP :** Internet of People

**IT :** Information Technology

**IAM :** Identity and Access Management

**IPS :** Intrusion Prevention System

**DDoS** : Distributed Denial of Service

**DoS :** Denial of Service

**SDPN** : Stacked-Deep Polynomial Network

**SMO** : Spider Monkey Optimization

**SGD :** Stochastic Gradient Descent

# ملخص

النمو المتزايد لأجهزة إنترنت الأشياء (IoT) قد حوّل بشكل كبير مختلف المجالات، بما في ذلك المنازل الذكية والرعاية الصحية والأتمتة الصناعية. ومع ذلك، فإن الاتصال المتزايد يقدم أيضًا نقاط ضعف أمنية كبيرة، مما يجعل تطوير أنظمة الكشف عن التسلل (IDS) الفعالة أمرًا ضروريًا. في هذه الدراسة، اقترحنا إطار عمل لنظام كشف التسلل باستخدام تقنيات التعلم العميق لتعزيز الأمان في بيئات إنترنت الأشياء من خلال التصنيف الثنائي.

يتضمن إطار العمل المقترح عدة نماذج للتعلم العميق، مثل المشفر التلقائي"AE" ، شبكة الالتفاف العصبية ذات البعد الواحد "CNN1D"، شبكة الالتفاف العصبية ذات البعدين"CNN 2D" ، الذاكرة طويلة وقصيرة الأمد"LSTM" ، الشبكة العصبية العميقة التغذوية"FFDNN" ، ونموذج الجمع بين "Vgg16" و "MobileNetV2"و ."DenseNet 121" تتضمن معالجة البيانات المسبقة تطبيع مجموعات البيانات، استخراج الميزات باستخدام المشفر التلقائي العميق، والموازنة باستخدام تقنية زيادة العينات الصغرى الاصطناعية ."SMOTE" تم تدريب النماذج وتقييمها باستخدام مجموعتي بيانات CICIDS2017وCICDDoS2019، التي تشمل مجموعة متنوعة من سلوكيات حركة المرور الشبكية وسيناريوهات الهجوم.

تشير النتائج التجريبية إلى أن معظم النماذج حققت دقة عالية، ولكن العامل الحاسم لاختيار النموذج الأفضل هو الاستدعاء، لأن في نظام الكشف عن التسلل، يكون معدل الإنذارات الكاذبة المنخفض أمرًا حاسمًا حيث تشكل الهجمات غير المكتشفة خطورة كبيرة. كان أفضل نموذج أداءً هو الجمع بين المشفر التلقائي ونموذجCNN 2D ، حيث حقق نتائج مثيرة للإعجاب بدقة بلغت 98.23%، واستدعاء 96.82%، ودقة 99.63%، ودرجة F1 بلغت 98.20% لمجموعة بيانات CICIDS2017، ودقة بلغت 98.43%، واستدعاء 99.38%، ودقة 97.59%، ودرجة F1 بلغت 98.48% لمجموعة بيانات .CICIDDOS2019 يوفر دمج المشفرات التلقائية مع الشبكات العصبية الالتفافية والتكرارية تمثيلًا شاملاً للميزات، مما يحسن قدرة النظام على تحديد التهديدات المعروفة والجديدة.

في الختام، تثبت هذه البحث فعالية نظام الكشف عن التسلل المعتمد على التعلم العميق في تعزيز أمان إنترنت الأشياء. تشير النتائج إلى أن هذه الأنظمة يمكن أن تقدم حلاً موثوقًا وفعالاً للكشف عن التسللات في الوقت الحقيقي، مما يمهد الطريق للتقدم المستقبلي في الأمن السيبراني لإنترنت الأشياء. يساعد هذا العمل في التخفيف من التهديدات السيبرانية في البيئات المتصلة بشكل متزايد، مما يبرز إمكانيات منهجيات التعلم العميق المتقدمة في تعزيز آليات الدفاع الشبكية.

**الكلمات المفتاحية** : نظام كشف التسلل (IDS) ، إنترنت الأشياء (IoT) ، المشفر التلقائي(AE) ، الشبكة العصبية الالتفافية (CNN)، الذاكرة طويلة وقصيرة الأمد (LSTM) ، الشبكة العصبية العميقة التغذوية (FFDNN) ، VGG16، MobileNet، DenseNet 121، CICIDS2017، CICDDoS2019 .

# Abstract

The growth of Internet of Things (IoT) devices has significantly transformed various domains, including smart homes, healthcare, and industrial automation. However, the increasing connectivity also introduces substantial security vulnerabilities, making the development of effective Intrusion Detection Systems (IDS) essential , in this study we proposed an Intrusion Detection System framework using deep learning techniques to enhance security in IoT environments through binary classification.

The proposed framework integrates several deep learning models, such as Autoencoder “AE”, One-Dimensional Convolutional Neural Network “CNN1D”, Two-Dimensional Convolutional Neural Network “CNN 2D”, Long Short-Term Memory “LSTM”, Feed-Forward Deep Neural Network “FFDNN” and “Vgg16” , “MobileNetV2” , DenseNet 121” combination Model, the data preprocessing involves normalizing datasets, feature extraction using a deep autoencoder, and balancing using the Synthetic Minority Over-sampling Technique “SMOTE”, the models were trained and evaluated using the CICIDS2017 and CICDDoS2019 datasets, which encompass a diverse range of network traffic behaviors and attack scenarios.

Experimental results indicate that most models achieved high accuracy, but the critical factor for choosing the best model is the recall because in an IDS a low false positive rate is crucial as undetected attacks pose significant dangers, the best-performing model was combining Autoencoder and CNN 2D Model, achieved impressive results with an accuracy of 98,23%, recall of 96,82%, precision of 99,63% , and F1-score of 98,20% for CicIDS2017 Dataset and accuracy of 98,43%, recall of 99,38%, precision of 97,59% , and F1-score of 98,48% for CicIDDOS2019 Dataset , the integration of autoencoders with convolutional and recurrent neural networks provides a comprehensive feature representation, improving the system's ability to identify both known and novel threats.

In conclusion this research demonstrates the efficacy of deep learning-based IDS in fortifying IoT security, The findings suggest that such systems can offer a reliable and efficient solution for real-time intrusion detection, paving the way for future advancements in IoT cybersecurity. This work aids on mitigating cyber threats, in increasingly interconnected environments, highlighting the potential of advanced deep learning methodologies in enhancing network defense mechanisms.

**Index Terms -** Intrusion Detection System (IDS) , Internet of Things (IoT) , Autoencoder (AE), Convolutional Neural Network (CNN) , Long Short-Term Memory (LSTM) , Feed-Forward Deep Neural Network (FFDNN) , VGG16 , MobileNet, DenseNet 121 ,CICIDS2017 , CICDDoS2019.

# Résumé

L'augmentation des appareils de l'Internet des objets (IoT) a transformé de manière significative divers domaines, notamment les maisons intelligentes, les soins de santé et l'automatisation industrielle. Cependant, la connectivité accrue introduit également des vulnérabilités de sécurité substantielles, rendant essentiel le développement de systèmes de détection d'intrusion (IDS) efficaces. Dans cette étude, nous avons proposé un cadre de système de détection d'intrusion utilisant des techniques d'apprentissage profond pour améliorer la sécurité dans les environnements IoT par le biais de la classification binaire.

Le cadre proposé intègre plusieurs modèles d'apprentissage profond, tels que l'autoencodeur "AE", le réseau de neurones convolutifs unidimensionnel "CNN1D", le réseau de neurones convolutifs bidimensionnel "CNN2D", la mémoire à long et court terme "LSTM", le réseau de neurones profondes feed-forward "FFDNN", et un modèle combinant "Vgg16", "MobileNetV2" et "DenseNet 121". La prétraitement des données implique la normalisation des ensembles de données, l'extraction de caractéristiques à l'aide d'un autoencodeur profond, et l'équilibrage à l'aide de la technique de suréchantillonnage des minorités synthétiques "SMOTE". Les modèles ont été entraînés et évalués en utilisant les ensembles de données CICIDS2017 et CICDDoS2019, qui englobent une gamme diversifiée de comportements de trafic réseau et de scénarios d'attaque.

Les résultats expérimentaux indiquent que la plupart des modèles ont atteint une haute précision, mais le facteur critique pour choisir le meilleur modèle est le rappel, car dans un IDS, un faible taux de faux positifs est crucial étant donné que les attaques non détectées posent des dangers significatifs. Le modèle le plus performant était la combinaison de l'autoencodeur et du modèle CNN2D, qui a obtenu des résultats impressionnants avec une précision de 98,23 %, un rappel de 96,82 %, une précision de 99,63 % et un score F1 de 98,20 % pour l'ensemble de données CICIDS2017, et une précision de 98,43 %, un rappel de 99,38 %, une précision de 97,59 % et un score F1 de 98,48 % pour l'ensemble de données CICIDDOS2019. L'intégration des autoencodeurs avec des réseaux de neurones convolutifs et récurrents fournit une

représentation complète des caractéristiques, améliorant la capacité du système à identifier à la fois les menaces connues et nouvelles.

En conclusion, cette recherche démontre l'efficacité des IDS basés sur l'apprentissage profond pour renforcer la sécurité des IoT. Les résultats suggèrent que de tels systèmes peuvent offrir une solution fiable et efficace pour la détection d'intrusion en temps réel, ouvrant la voie à des avancées futures dans la cybersécurité des IoT. Ce travail contribue à la réduction des menaces cybernétiques dans des environnements de plus en plus interconnectés, mettant en évidence le potentiel des méthodologies avancées d'apprentissage profond pour améliorer les mécanismes de défense réseau.

**Mots-clés :** Système de détection d'intrusion (IDS), Internet des objets (IoT), Autoencodeur (AE), Réseau de neurones convolutifs (CNN), Mémoire à long et court terme (LSTM), Réseau de neurones profondes feed-forward (FFDNN), VGG16, MobileNet, DenseNet 121, CICIDS2017, CICDDoS2019.

# Introduction

The Internet of Things (IoT), through its ability to interact with real-world applications via the internet, has revolutionized our engagement with the physical world. This technological advancement has integrated elements such as computers, sensors, urban infrastructure, and even communities into an intelligent ecosystem. IoT spans across various sectors including education, health, agriculture, defense, and commerce, offering innovative solutions to enhance quality of life and streamline processes. Furthermore, IoT is characterized by its connectivity potential, utilizing universally recognized protocols for information transmission, and its flexibility, capable of addressing a wide range of specific needs. Its pervasive presence in our daily lives also earns IoT its other name, Internet of People (IoP)

**Problematic:**

The rapid pace of technological advancement and the continued increase in connected devices around the world, exemplified by the development of IoT, Big Data and cloud computing, has reinforced the need for robust security for networks. With increased use of online services, network security has become crucial, as any breach can compromise the entire system. Despite traditional protection measures like firewalls and encryption, cyberattacks are becoming increasingly sophisticated , These have created major security vulnerabilities that increase the need to enhanced IoT Network security and highlight the urgency of developing efficient intrusion detection systems (IDS) [1],these systems play a critical role in preventing unauthorized access, protecting information and communications systems, and reliably detecting threats, whether known or new, in order to maintain the integrity, confidentiality and the availability of data on the networks.

**Primary aims work of this study** :

To mitigate attacks such as botnets and DDoS exploiting these vulnerabilities, deploying an effective defense mechanism against cyberattacks (intrusion detection systems (IDS)) is a potential solution. IDSs detect signs of cyberattacks by continuously monitoring network traffic generated by IoT devices. Attack detection is classified into two methods: signature-based and anomaly-based. The first uses rules created from known attack patterns, while the second trains a model with normal system behaviors to detect anomalies. However,

these methods model a limited number of system behaviors, insufficient for heterogeneous IoT networks, which degrades detection accuracy.

Artificial intelligence (AI) has greatly contributed to improving the learning skills of computers and machines, enabling autonomous learning from large amounts of data with reduced human intervention. This progress has benefited intrusion detection systems, which have been able to adapt to this new form of intelligence ,among the subfields of AI, machine learning (ML) and deep learning (DL) have been particularly useful in the development of successful intrusion detection systems. In machine learning, classification and detection of network traffic is based on manually identified characteristics. In contrast, deep learning, through its neural networks, is able to automatically extract relevant features from data and then use this information for classification and detection[2]. This method offers the advantage of improving detection accuracy compared to traditional machine learning.

Motivated by the success of deep learning, and based on various learning approaches and techniques, many models have been and continue to be developed to create an effective intrusion detection system. In this study, we attempt to develop more efficient detection models. This work proposes an intrusion detection system based on deep learning which uses some deep learning algorithms and also combinations of Auto encoder (AE) with each of these algorithms, One-Dimensional Convolutional Neural Network (1D-CNN), (2D-CNN) and long short-term memory (LSTM), Feed-Forward Deep Neural Network (FFDNN), MobileNetV2, Visual Geometry Group (VGG16).

This study evaluated the model using the CIC-IDS2017, CICDDOS 2019 datasets, and compared the results to studies that use other learning algorithms and to other datasets. Our results demonstrated that the integration of two deep learning algorithms will improve the detection rate and accuracy, making the model more accurate and resistant to threats and attacks.

The first objective of this work is to present the key concepts of IDS and deep learning. The second objective is to provide recent trends and observations on intrusion detection systems using deep learning. We review the published works, their advantages, disadvantages, datasets, techniques and evaluation metrics. We discuss what has been done and what could be done to improve IDS using new deep learning techniques. Finally, future trends are described.

The study is structured as follows: Section 1 presents a Background to the concepts of IoT security. Section 2 includes a well-detailed description of intrusion detection systems: their

different types, their working principles, their advantages and reviews the relevant articles used in the study. Our hybrid models and the description of the datasets used in the development of the model are structured in section 3. Section 4 describes the experimental design and evaluation of the model, while section 5 presents future trends and concludes this work.

# I Chapter I : Background and IoT security

## I.1 Introduction

Internet of Things (IoT) technology is prospering and entering every part of our lives, be it education, home, vehicles, or healthcare. As the number of connected devices increases, many challenges arise with IoT technology, including security requirements, the management of which passes the second plan for reasons of cost, form and power. Researchers have proven many security attacks against IoT devices. In this section, we review some of these attacks and the various security solutions proposed by researchers to detect breaches in various applications of Internet of Things networks.

## I.2 IoT Definition

Today, we have an emerging paradigm that allows for the connection between physical objects and the information capabilities of the Internet, This is what constitutes the Internet of Things (IoT) at scale. Recently, researchers predict that total number of devices connected to internet would exceed 6.4 Billion by 2016 and this number is expected to reach 20.8 billion by 2020 [3]. The list of such devices includes iPhone, iPad, iWatch, Smart TVs and many more devices. The European Research Cluster on IoT (IERC) defines IoT as:
***"a dynamic global network infrastructure with self-configuring capabilities based on standard and inter-operable communication protocols where physical and virtual things have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network"***[3]

## I.3 IoT Architecture

There is no universally agreed upon architecture for the Internet of Things. Many researchers have proposed different architectures. One commonly discussed approach is the three- and five-layer architecture, which was introduced early in research in this area. It consists of three layers: perception, network, and application layers, as shown in Figure 1:

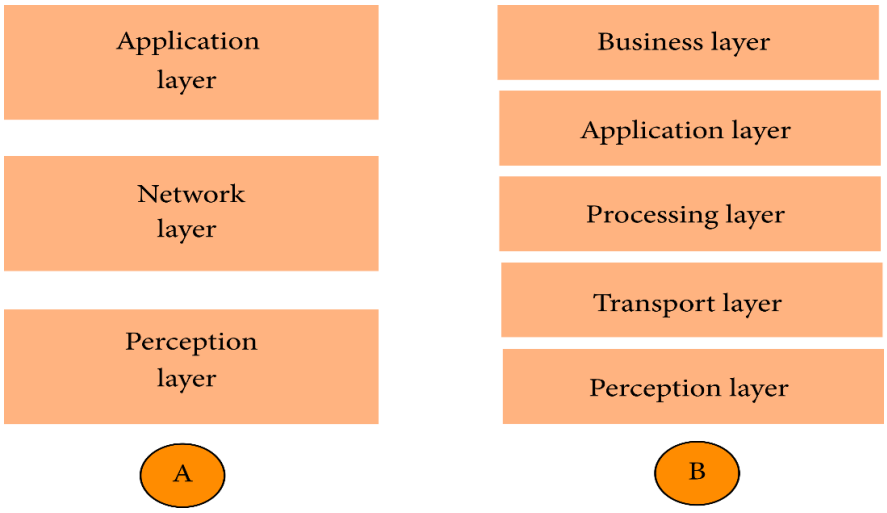



**Figure 1:Architecture of IoT (A: three layers) (B: five layers)**

**The perception layer** is the physical layer, which has sensors for sensing and gathering information about the environment. It senses some physical parameters or identifies other smart objects in the environment.

**The network layer** is responsible for connecting to other smart things, network devices, and servers. Its features are also used for transmitting and processing sensor data.

**The application layer** is responsible for delivering application specific services to the user. It defines various applications in which the Internet of Things can be deployed, for example, smart homes, smart cities, and smart health.

**The three-layer architecture** serves as a foundational concept for understanding the Internet of Things. However, it is not comprehensive enough to address the intricacies explored in IoT research, which often delve into more nuanced aspects. Consequently, there are several additional layered architectures proposed in the literature. One such model is the five-layer architecture, which expands upon the three-layer model by incorporating processing and business layers. In this framework, the five layers are perception, transport, processing, application, and business layers (depicted in Figure 1). While the perception and application layers retain similar roles as in the three-layer architecture, the functions of the remaining three layers are further delineated[4].

**The processing layer** is also known as the middleware layer. It stores, analyzes, and processes huge amounts of data that comes from the transport layer. It can manage and provide a diverse set of services to the lower layers. It employs many technologies such as databases, cloud computing, and big data processing modules.

**The transport layer** transfers the sensor data from the perception layer to the processing layer and vice versa through networks such as wireless, 3G, LAN, Bluetooth, RFID, and NFC. Creation of reliable and effective business models, the management of business regulations, and the deployment of optimal investment are some of the major responsibilities of the business layer in an IoT network. This is based on the fact that business models are created by evaluating the effectiveness of current applications and services. The use of technology and the ways in which services are presented to customers/clients may hence be used to measure the success of the IoT system [5]

## I.4 Importance and Application Domains of the IoT

The Internet of Things (IoT) is poised to revolutionize the global technology landscape, with its applications extending to billions of connected smart devices that utilize vast amounts of data from our daily lives. This technological advancement, marked by the emergence of new wireless networks, high sensors, and superior capabilities, promises to simplify our lives and bring immense value. IoT applications are increasingly becoming an integral part of our everyday lives, with their rapid evolution and the development of IoT suites. However, this integration comes with significant security, privacy, and agreement challenges that can impact various aspects, from applications to the environment and commerce. To address these concerns, it is crucial to implement robust security measures tailored to each IoT domain, with a focus on function-based security solutions. Figure 2 represents the various application domains were IoT systems are implemented.

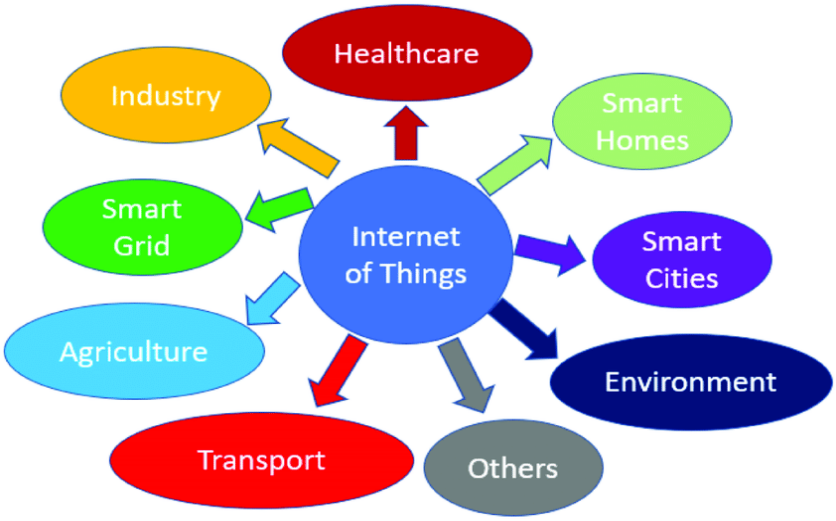



**Figure 2:Application Domains of IoT** [6]

### I.4.1 Smart Home

Modern homes are increasingly equipped with smart technology, including advanced lighting, refrigerators, washing machines, air conditioning units, electricity meters, alert devices, and CCTV systems. These homes are designed with security and safety in mind, utilizing smart cameras, sensors, smart locks, and alarm systems. The smart home technology allows for remote control and management of these devices via the internet, ensuring that the devices are securely protected with password-protected access[6].

### I.4.2 Smart City

The concept of a smart city, driven by IoT technologies across various sectors, is gaining traction. However, a significant challenge remains in ensuring the confidentiality and trust among all stakeholders in this domain [7].

### I.4.3 Industrial IoT

IoT that allows for remote monitoring, diagnosis, and control of physical processes and show output in real time. Industrial IoT and its goods, on the other hand, are closely linked to the internet of things. Many security issues affect digital entities, including confidentiality, anonymity, and trust. They also pose problems such as output standardization of the structure and public aspects. Due to a scarcity of resources the industrial IoT design necessitates low-cost, low-power infrastructures that are completely integrated[6].

### I.4.4 Smart Grid

A smart grid is an intelligent electrical power system that consists of a network of electrical transmission lines, transducers, and substations that carry electricity from a power plant to homes and businesses. The main areas of concern that need to be addressed are verification, privacy, reliability, availability, and availability [6] .

### I.4.5 Agriculture

The continuing increase in global population has increased demand for agricultural products. However, the migration of young people to major cities hampers the human resources needed for agricultural development. The Internet of Things (IoT) and related technologies can play a crucial role in automating agricultural processes and meeting food demand.

### I.4.6 Health care IoT

IoT mobile devices provide a range of benefits to patients and health professionals. For example, the Internet of Things allows health workers to monitor patients remotely. These devices can automatically collect patients' lives, such as arterial pressure, heart rhythm, temperature and more.

## I.5 IT security

Computer security is all the hardware and software means implemented to protect any computer system and its data against any violation, leak, publication of private information or attack[i] thus allowing the system to operate normally and to ensure the availability of the expected services.

Objectives of Security information aims to protect sensitive data and information from unauthorized access and manipulation. This protection includes many aspects, including:

- **Confidentiality:** means protecting data from unauthorized access by unauthorized persons. The importance of confidentiality is to protect sensitive and private data from unauthorized access and illegal use.
- **Integrity**: is about ensuring that the people who get the data are the people who should get it. This is done by identifying the persons authorized to access the data and defining their powers and responsibilities.
- **Authenticity:** means ensuring that data has not been tampered with or changed in an unauthorized manner. The importance of credibility is to ensure the validity and accuracy of data and to avoid errors and manipulation.
- **Non-repudiation**: This is about making sure that people who interact with data cannot deny that they did so. This is done by documenting processes, recording activities, and verifying the identity of people interacting with the data.

## I.6 Security Attacks in IoT

Within this part,We review how to address typical threats to IoT networks, and list distinct attacks that can be launched on an IoT network, with an emphasis on network-based attacks and potential risks to end users.

- **Botnet :** IoT botnets are attractive to botnet masters because they typically come with insecure default settings that allow malicious code to be injected into victim machines that allow unauthorized access to management systems and execute insecure code that is exploitable by botnet masters.
- **Déni de service (DoS) / Déni de service distribué (DDoS) :**These attacks aim not at accessing information on a remote machine but at crippling a service or an entire network, rendering the user unable to access resources. The two main examples are ping-flood, or the massive sending of emails to saturate a mailbox (mailbombing) [7]

Figure 3 shows an example IDS for DDoS attacks :

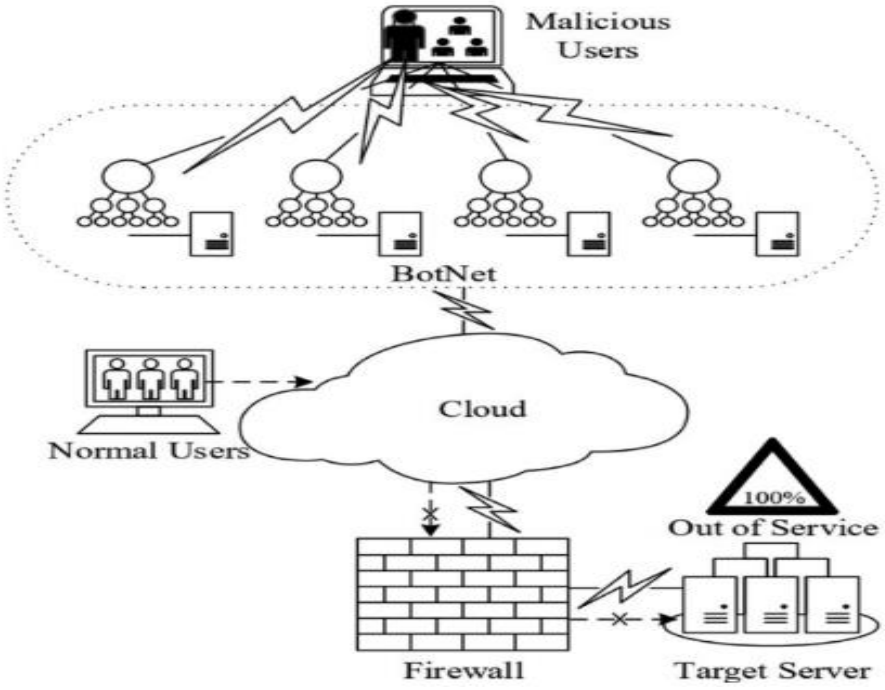



**Figure 3:An example IDS for DDoS attacks** [8]

- **Remote Recording:** Cybercriminals are recording IoT users' conversations by hacking into smart cameras. To combat this threat, companies must establish strict cybersecurity policies, ensure sensitive data is encrypted, and review systems regularly. Emerging technologies, big data and artificial intelligence can also be used to enhance security against such threats.
- **Ransomware:** Ransomware is being imposed as one of the most dangerous cyber threats. In this scenario, the attacker uses malware to sort through important data and, after passing through a copy of the text, deletes the data to make it accessible again.
- **Identity and Data Theft:** Cybercriminals provide sensitive information such as personal data, credit and debit card numbers, as well as email addresses that may be requested. These identity-related attacks are specific to companies using technologies such as connected watches, smart computers, and home devices. Data breaches are common and frequent in business systems[6].
- **Man in the Middle:** The cyber pirate, also known as the "Man in the Middle," will intercept communications between two distinct IoT devices. By taking control of this communication, attackers can send unauthorized information to these specific IoT devices. These attacks are particularly common in industrial and medical instruments[6].

### I.6.1.1 Real-world Examples of IoT Security Breaches

The vulnerabilities inherent in IoT devices have led to several high-profile security breaches:

1. **Mirai Botnet Attack (2016)**

   The Mirai botnet compromised numerous IoT devices, such as cameras and routers, to execute a massive Distributed Denial of Service (DDoS) attack. This incident disrupted major websites including Twitter, Netflix, and Reddit, highlighting the vulnerabilities present in IoT devices and the potential scale of such attacks .

2. **St. Jude Medical Cardiac Devices (2017)**

   Researchers discovered vulnerabilities in St. Jude Medical's cardiac devices, which could potentially allow hackers to interfere with the device's operation, endangering

patients' lives. This incident underscores the critical importance of security in medical IoT devices .

3. **Smart Home Devices Hacking (2019)**

Several incidents involved hackers exploiting security flaws in smart home devices, such as cameras and baby monitors, to spy on users. In some cases, hackers communicated directly with people in their homes, demonstrating the severe privacy risks associated with poorly secured IoT devices .

## I.7 How to protect computer systems

To ensure the security of information systems, it is essential to identify what needs to be protected, potential risks and system vulnerabilities. From this analysis, it is possible to determine the mechanisms and means necessary to strengthen security, taking into account constraints such as costs and complexity. Among the strategies commonly used to secure information systems, we find in particular:

- **Firewall**: A firewall is the first line of defense. It acts as a filter between the internal and external network, controlling incoming and outgoing traffic according to pre-defined rules. By preventing unauthorized access while allowing legitimate flow, it limits exposure to external threats.
- **Data Encryption:** Data encryption is an essential security measure. By making data unreadable to unauthorized persons, even if they gain access to it, it ensures the confidentiality and integrity of sensitive information.
- **Identity and Access Management (IAM):** Strict management of user accounts, privileges, and access to resources is essential to reduce risk. By ensuring that only authorized individuals have access to sensitive data, IAM reduces the chance of intrusion.
- **Regular updates and patch management:** Software and systems are constantly exposed to new security vulnerabilities. Therefore, keeping all software and systems updated with the latest security patches is crucial to reducing the risk of exploitation by attackers.
- **Regular backups:** Regular backups provide insurance in the event of a security incident or disaster. By having regularly updated copies of data and systems, information can be quickly restored when needed, reducing loss and downtime.

- **Employee training and awareness**: Users are often the weakest link in security. Regular training on IT security best practices and awareness of current threats is therefore essential to reduce the risk of social engineering attacks and improve an organization's overall security posture.
- **Intrusion Prevention Systems (IPS):** Intrusion prevention systems go beyond mere detection with their ability to automatically block detected malicious activity. This ability to act in real time significantly reduces the reaction time required in the event of an attack, thus enhancing system security.
- **Intrusion Detection Systems (IDS):** These systems are deployed to monitor network traffic or system file activities in real time. By detecting suspicious behavior or signatures of known attacks, they provide early warning of malicious activity.

## I.8 IoT Intrusions

In the Internet of Things, intrusions and threats can arise through three groups of threat agents. On the physical couch, we can avoid physical attacks, hand-grabbing attacks and pursuing users. Through the Internet, we can avoid basic Internet attacks (e.g., DoS, doxing, people-in-the-environment, covert communication), Internet attacks (e.g., hacking and snooping, DDoS), routing attacks, and attacks on network ports. On the application surface, we are threatened by attacks such as malware, spyware, SMS, DoS attacks, viruses, spoofing, secret calls, and others. Distributed attacks, in particular, can have a significant impact, and DDoS attacks are widely used. Table 1 lists different types of attacks that can affect each sofa, as well as attacks that use multiple sofas. Additionally, the list of features represented by each bed contains what has been certified for intrusion detection. From all evidence, the properties of a perceptual couch/physical couch relate to the properties of signal transmission and reception, and on the Internet, the properties relate to the properties of packets and data streams identified through inspection, as well as the properties of the application The board is specific protocols examined by IDS, such as HTTP, FTP, and SSH CANbus, Modbus, other SCADA protocols, etc. Triple-A attacks can compromise data confidentiality, integrity, availability, and confidentiality, causing harm to organizations, individuals, and countries[9].Table 1 describes the different types of attacks and intrusions affecting each layer of the IoT architecture :

**Table 1:**Types of attacks and intrusions affecting each layer of the IoT architecture [9]

| Layer | Type of attack | Example of feature bases |
|---|---|---|
| Perception/ physical layer | Physical attacks, node capture, user tracking, social engineering to access devices, side-channels, radio frequency jamming, sleep deprivation attack | PHYL specific features, such as signal voltage, received signal strength indicator, signal to noise ratio, bitstream patterns |
| Network layer | DoS, DDoS, spoofng, trafc analysis, wormhole, man-in-the-middle, eavesdropping, sybil, cloning, sinkhole, hacking, viruses, identity theft | Network specifc features, such as source IP address, destination IP address, transaction bytes, time to live, packets retransmitted, packets dropped, packet size |
| Applications layer | Malware, spyware, ransomware, bufer overfow, DoS, DDoS, viruses, botnets, phishing, SQL injection, spoofing, privilege escalation, eavesdropping | Application specific features, such as transaction protocol, number of connections, rate of connections, connection duration, PDU sequence number, login attempts, status fags |
| Multi-layer | Man-in-the-middle, trafic analysis, side-channels | (all) |

# II Chapter II:

# Intrusion detection systems Using DL

## II.1 Motivation

IoT devices typically operate in unmonitored environments, making them susceptible to unauthorized access by intruders. Covert listening can enable access to private information transmitted through the communication channel, as IoT devices are often connected via wireless networks. Beyond these security concerns, IoT devices are constrained by limited energy and processing power, preventing them from integrating advanced security features. Therefore, an additional layer of protection should be integrated into IoT networks to safeguard IoT-based organizations against cyber threats. The integration of intrusion detection systems (IDS) specifically designed for IoT is crucial for enhancing the security of these devices and the data they manage. Deep learning techniques have recently become more reliable in a popular application for detecting network attacks, such as IoT networks. Deeply trained systems in IoT environments will primarily pick up anomalous and benign behaviors , offering a promising approach to overcome the limitations of traditional IDS methods, thus providing a more robust and adaptable solution for detecting and mitigating cyber threats in environments.

## II.2 Intrusion detection system

The idea of intrusion detection systems (IDS) was first introduced in 1980 by James Anderson in his article "Computer Security Threat Monitoring and Surveillance." It described how reviewing audit logs could be used to identify unauthorized or malicious actions. During the 1980s, many systems managers adopted the use of IDS. However, these systems suffered from vulnerabilities and could not detect zero-day attacks, while consuming a lot of resources to continually scan the network. Towards the end of the 1990s, a new detection approach was developed to deal with the increase in attacks. This method, known as anomaly detection, aimed to identify unusual behaviors or activities in a system to trigger an alert. However, network diversity between the 1990s and 2000s led to a large number of false positives, prompting many administrators to abandon the use of IDSs due to their imprecision. Fortunately, with improvements in networking and increased processing power of network components, as well as the emergence of machine and deep learning, IDSs have become an essential part of IT security. Many machine and deep learning techniques have been tested for the development of IDSs. With deep learning, new solutions can be considered to reduce the problem of false positives and improve detection accuracy[10].

An intrusion detection system (IDS) is comprised of a software or hardware application that examines network traffic for malicious behavior or policy violations, providing alerts when suspicious activity is detected. These alerts are often transmitted to a system or centralized administrator through the security information and event management (SIEM) system, which integrates data from multiple sources and applies alarm filtering mechanisms to differentiate genuine alerts from false ones. IDSs can take many forms, ranging from antivirus software to layered monitoring systems that monitor all network traffic. Network intrusion detection systems (NIDS) and host-based intrusion detection systems (HIDS) are the most general categories. The ultimate goal of an IDS is to protect an information system against unauthorized access and potential cyberattacks, such as manipulation, illegal access or other forms of attacks[8].

Depending on the strategy used to detect intrusions,people who steal data or cause intrusions are called intruders,the main intruders are the Criminal (legitimate/internal user), the Deceiver (unauthorized/external user) and the Covert User (internal/external). Existing literature has described different types of IDS based on different criteria such as information source, analysis process, detection technique, learning method, and knowledge base. IDS can be classified into two main groups. One group uses cross-checking of monitored events with a database of intrusion techniques or another group analyzes normal behavior and reports abnormal events, Table 2 describes the Evolution and Classification of Intrusion Detection Systems (IDS):

**Table 2**:Evolution and Classification of Intrusion Detection Systems (IDS)

| **Aspect** | **Description** |
|---|---|
| Introduction of IDS | Introduced in 1980 by James Anderson in 'Computer Security Threat Monitoring and Surveillance'. Described using audit logs to identify unauthorized actions. |
| 1980s | Widespread adoption of IDS by systems managers. These systems were resource-intensive, struggled with zero-day attacks, and consumed a lot of resources. |
| Late 1990s | Development of anomaly detection to identify unusual behaviors or activities, triggering alerts. |
| Challenges in 1990s-2000s | High false positive rate due to network diversity, leading many administrators to abandon IDSs due to imprecision. |

| | |
|---|---|
| Improvements | Advances in networking, processing power, and the emergence of machine learning and deep learning improved IDS effectiveness. |
| Modern IDS | Essential part of IT security, incorporating ML and DL to reduce false positives and improve detection accuracy. |
| Components of IDS | Software or hardware applications examining network traffic for malicious behavior or policy violations, with alerts sent to a centralized administrator via SIEM. |
| Types of IDS | Network Intrusion Detection Systems (NIDS): Monitors all network traffic.<br>Host-Based Intrusion Detection Systems (HIDS): Monitors individual hosts. |
| Goals of IDS | Protect information systems against unauthorized access and cyberattacks, including manipulation, illegal access, or other attacks. |
| Types of Intruders | Criminal: Legitimate/internal user.<br>Deceiver: Unauthorized/external user.<br>Covert User: Internal/external user. |
| Classification Criteria | Information Source: Cross-checking monitored events with a database of intrusion techniques.<br>Analysis Process: Analyzes normal behavior and reports abnormal events.<br>Detection Technique: Signature-based, anomaly-based, etc.<br>Learning Method: Supervised, unsupervised, etc.<br>Knowledge Base: Rules, patterns, etc. |

Figure 4 illustrates the different types of intrusion detection systems (IDS) that can be used to detect network intrusions in the IoT network.

There are three stages in the operation of an IDS,in the first stage (monitoring stage), network-based sensors are used to monitor the environment,feature extraction methods or pattern identification methods are used in the second stage, which is the analysis stage, Anomaly detection and misuse intrusion detection constitute the detection stage,the purpose of an IDS is to capture a copy of data traffic from information systems and analyze this copy to identify potentially harmful activities [11] to ensure adequate data accuracy and smooth operation of processes. which are associated with it.

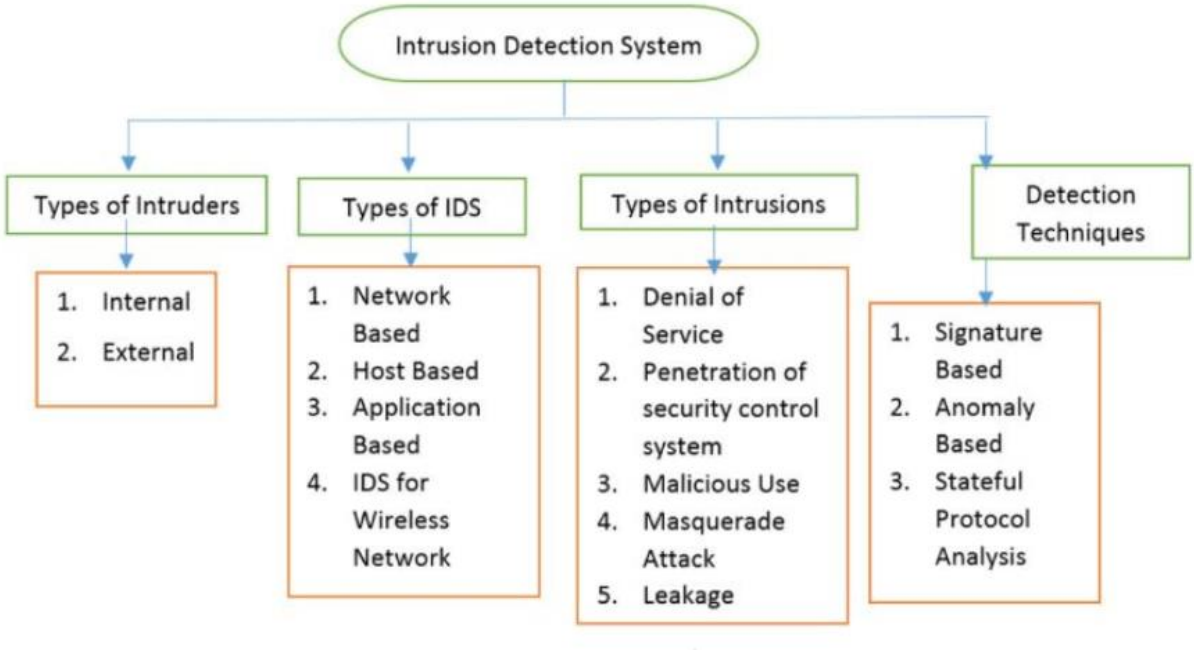



**Figure 4:Types of IDS [11]**

## II.3 Information source based IDS

Intrusion Detection Systems (IDS) can be categorized into three main types based on their information source: host-based, network-based, and hybrid , Figure 5 shows the classification of intrusion detection systems*:*

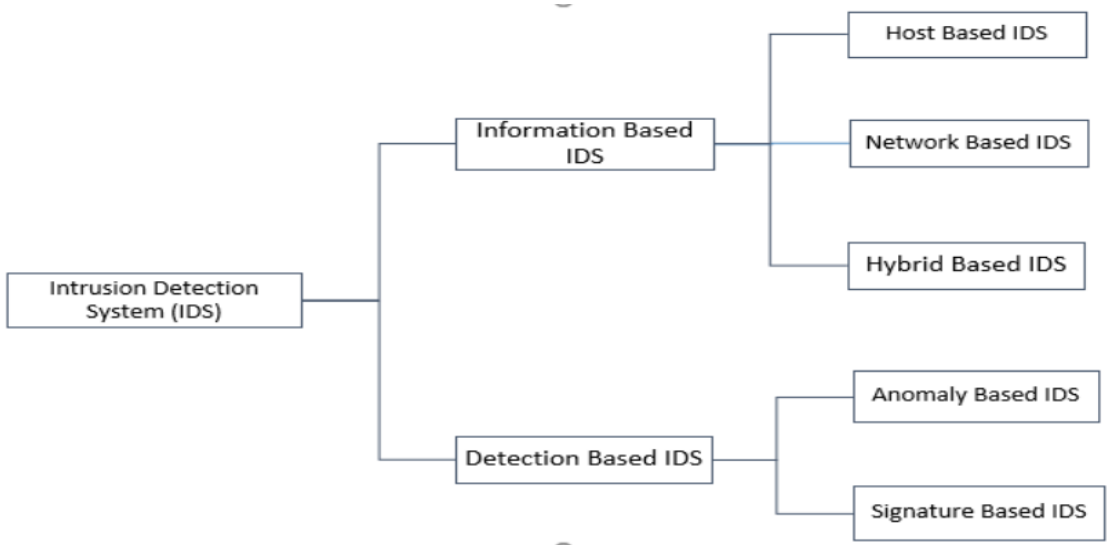



**Figure 5:Classification of intrusion detection systems [5]**

### *II.3.1* Network Intrusion Detection System (NIDS)

The NIDS analyzes all traffic sent by all devices on the network to a scheduled point in the network. By observing and comparing traffic flowing across the entire subnet to known attacks, it can detect and stop attacks before they occur. The administrator can receive an alert after an attack or abnormal behavior has been identified. NIDS can be used on a subnet containing firewalls to check if they have been compromised [11]. In Summary Network-based intrusion detection systems (NIDS) examine network-related information, such as network packets, network protocol broadcast domains, traffic volume, and IP addresses, to detect attacks.

### II.3.2 Host Intrusion Detection System (HIDS)

Host-based intrusion detection systems (HIDS) are designed to monitor and detect intrusions on a local host by leveraging the host's file system, system calls, and network events. They are specifically suited to detecting attacks targeting the system on which they are installed. Additionally, these systems, also known as real-time HIDS, are installed on computers or stand-alone devices. When they identify suspicious or malicious activity, they alert the appropriate administrator. They operate by exclusively monitoring packets entering and leaving the device, comparing the current state of the system with that previously recorded, and triggering an alert in case of modifications or deletions detected in the scanning system files. Critical machines with fixed configurations provide a typical example of HIDS application. The analyzes performed by these systems rely primarily on metrics specific to the host environment, such as a computer system's log files, which serve as the basis for making informed decisions [11],in short, real-time HIDS analyze characteristics of the host environment to identify potentially harmful activities, thereby focusing on specifically protecting the host system on which they are deployed.

### II.3.3 Hybrid Intrusion Detection System (Hybrid IDS)

Hybrid intrusion detection systems (HIDS) merge the capabilities of multiple types of IDS, integrating both host agents and network data to provide a holistic picture of the network system [11]. This hybrid approach enables more effective intrusion detection compared to traditional methods, thanks to its ability to combine the advantages of different monitoring techniques, including real-time detection.

## II.4 Detection based IDS

In principle, the two primary methods of intrusion detection systems are signature-based and anomaly-based.

### II.4.1 Signature based intrusion detection

In signature-based intrusion detection systems (SIDS), signature or pattern matching techniques are used to identify previous intrusion. These methods typically provide high accuracy in detecting known intrusions and a low false alarm rate (FAR) because an alarm is generated only if a matching pattern is found. However, they require frequent signature updates to ensure effective detection , SIDS face challenges when identifying zero-day attacks because there is no matching signature in the database until the new attack signature is retrieved and recorded[12],Furthermore SIDS is a resource-intensive approach due to maintaining a large signature database and comparing potential interventions.

### II.4.2 Anomaly based intrusion detection

An anomaly-based IDS also referred to as behavior-based detection, mimics the behavior of users, computer systems, and networks. The intrusion detection system uses a variety of ways to build a baseline from typical behavior patterns,any major departure of the examined pattern from the norm is marked as an anomaly,the term “intrusion” refers to a device’s activity that deviates from its usual behavior[13] , AIDS has the ability to detect unknown or previously unencountered attacks due to its continuous learning capability, AIDS is developed in two phases: training and testing,during the training phase, the usual traffic profile is used to learn the normal behavior pattern, followed by a new dataset in the testing phase to establish the system's ability to generalize to unanticipated intrusions[12] .

## II.5 Security challenges in IoT and IoT IDS

IoT security is becoming increasingly critical as IoT networks expand, integrating a multitude of services and users. Smart objects, when integrated into smart environments, significantly improve their efficiency. However, IoT security vulnerabilities can have dramatic consequences in vital sectors like healthcare and manufacturing, which support critical functions in smart environments. Without strong security systems, IoT environments are vulnerable to security threats. Information security in IoT systems requires special attention, especially regarding confidentiality, integrity and availability, calling for further research in this area.

Security challenges, complexity, and compatibility of IoT environments pose significant obstacles to realizing smart environments in the real world. The services provided by these environments can be compromised by attacks such as DDoS and DoS attacks on IoT networks. IoT presents a range of security challenges from multiple angles, including vulnerabilities related to the IoT communications protocol.

On the physical side, challenges include physical damage, hardware failure, and power constraints, at the network level, challenges can include DoS attacks, sniffing, gateway attacks, and unauthorized access. At the application level, challenges encompass malware attacks, application vulnerabilities, and software bugs. With the rapid expansion of connected devices in the IoT, communications security is becoming a major challenge, highlighting the need for future research to improve existing IDSs to be more accurate, capable of detecting a variety of intrusions with fewer false alarms and with additional capabilities. Detecting attacks masked by evasion techniques represents a crucial challenge for SIDS and AIDS. As these evasion techniques become more sophisticated, it is essential that IDSs are able to recognize either the original signature of an attack or create new signatures to cover the changes made. Research is still needed to assess the robustness of IDS against various evasion techniques, including the effectiveness of SIDS regular expressions against simple mutations such as whitespace manipulation, while remaining ineffective against certain obfuscation techniques used by hackers, such as encryption and packing[11].

## II.6 IDS and Deep Learning for IoT Security

Intrusion detection systems (IDS) mainly consist of two modules: a data collection module and an intrusion analysis module. Input data from IoT systems is examined in the data collection module to identify interaction behaviors using various machine learning techniques in the analytics module. Deep learning has demonstrated exceptional performance in various fields, proving to be a more effective alternative to traditional machine learning. Additionally, IDSs that adopt deep learning have shown promising results. Deep learning based methods are more relevant and suitable for traffic analysis to distinguish normal traffic from abnormal traffic in an IoT environment. They are also able to intelligently predict new attacks, unlike already known threats , Figure 6 presents IDS using deep learning models in the IoT environment .

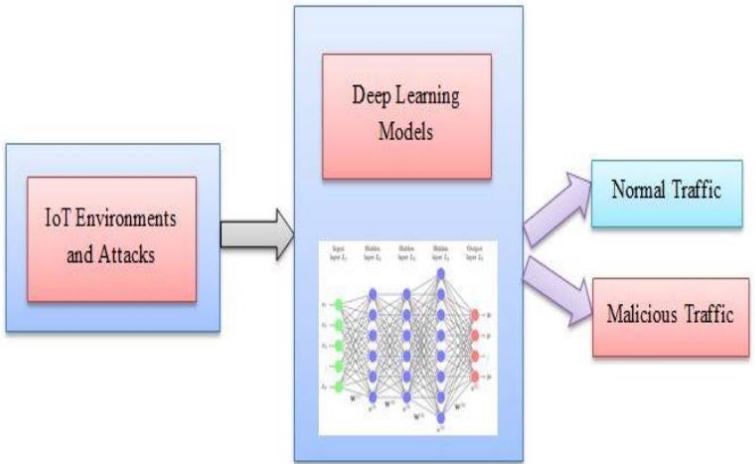



**Figure 6:IDS using deep learning models**

Machine learning and deep learning are two key disciplines of artificial intelligence (AI), are distinguished by their approaches and applications. Machine learning, based on statistical methods, allows machines to learn without explicit instructions, facilitating applications such as image recognition, natural language processing and forecasting.
On the other hand, deep learning, a subfield of machine learning, uses neural networks to simulate human cognition, efficiently processing unstructured data and accomplishing complex tasks like object recognition, voice recognition and autonomous driving, these technologies have the potential to revolutionize various sectors, including healthcare, finance and commerce, through their ability to continually learn and improve but, Deep neural networks, the basis of deep learning, enable hierarchical feature learning, suitable for large and complex datasets.

Additionally, deep learning offers higher abstraction through its many learning layers, outperforming traditional models as data size increases, this constant evolution is driving the development of more sophisticated, faster and more accurate algorithms, essential for organizations seeking to maintain a competitive advantage[14] this is why in our study we decided to use deep learning techniques.

## II.7 Comparison between deep learning and traditional machine learning

Machine learning (ML) and deep learning (DL) differ significantly in their approach, hardware requirements, conception time, methodology, and applications. Human engagement is essential to achieving the desired results with machine learning, as it requires continuous interaction to adjust and optimize models. In contrast, deep learning, while more difficult to implement initially, requires little intervention after deployment, thanks to its ability to learn and improve autonomously with big data.

In terms of hardware, machine learning algorithms are generally less complex and can be run on standard computers, while deep learning systems require robust hardware resources and graphics processing units (GPUs) to handle the complexity. calculations necessary for training neural networks. The growing demand for computing power has led to an increase in the use of GPUs, which offer high-bandwidth memory and an ability to hide memory transfer latencies through their thread parallelism.

The time required to install and use these technologies also varies: machine learning systems can be set up quickly but may not produce results as good as they could be, while deep learning systems take more time to configure but can provide immediate results that improve in quality as more data becomes available. Another distinctive aspect is the method of approach: machine learning relies on organized data and uses traditional techniques like linear regression, while deep learning is designed to process large amounts of unstructured data using neural networks.

The applications of these technologies are varied: machine learning currently finds application in areas such as email, banking and medical offices, while deep learning allows the development of more complex and autonomous programs, such as self-driving cars. and surgical robots. Finally, regarding data usage, machine learning algorithms generally use less data than deep learning ones, where data quality is crucial, Deep learning algorithms require significant amounts of data to train neural networks, but they can learn and improve autonomously as more data is processed[14].

## II.8 Deep learning for IoT

Deep learning (DL) is a cutting-edge field of machine learning (ML) with powerful analytical capabilities, used for applications such as computer vision, bioinformatics, and natural language processing. It is known to exhibit significant performance improvement over some ML algorithms and has recently been used in some IoT applications in edge/mist computing. These applications require the transfer of substantial volumes of data over the network, and DL models the learning process using multiple layers of neurons forming a heavier artificial neural network (ANN).

In fact, the term "deep" refers to the large number of layers through which data is transformed. Additionally, DL offers better performance with large data than ML, and can work with new features that can be used to solve problems without human interactions[15]. When training systems, it is possible to use supervised or unsupervised learning techniques,these techniques allow models to be applied to supervised learning tasks, such as classification, as well as unsupervised tasks, such as pattern analysis , Deep learning models rely on extracting higher level features from lower level features to obtain a stratified representation of the input data through an unsupervised learning approach across different levels entities.

Classification of concepts and theories is achieved by learning different layers of data representations, corresponding to different levels of data absorption. Some of the deep learning frameworks mentioned in the literature include deep belief networks (DBN), deep/stacked autoencoder networks (DAE/SAE), and convolutional neural networks (CNN) , these frameworks have been employed in various fields, such as natural language processing, speech recognition, audio recognition, object recognition and detection, and computer vision[16].

Deep learning is a branch of machine learning algorithms that uses multiple layers of nonlinear processing nodes for feature extraction and transformation, learns in an unsupervised manner (such as pattern analysis), and/or supervised (like classification), and learns several levels of representations linked to different levels of abstraction, representing a hierarchy of concepts.

### II.8.1 Deep learning classification

Generally, machine learning algorithms are categorized into supervised, unsupervised, and reinforcement learning. Deep learning (DL) is a subfield of machine learning that carries out representation learning through multilayer transformation, thereby generating more accurate results for detection and prediction tasks. Below are these categories based on the specific purposes for which each category was developed[16]:

- **Deep unsupervised or generative learning networks**

These networks aims to capture the high-order correlation of unlabeled input data for recognition purposes or synthesis of models. When used to characterize distributions common statistics of observed data and their associated classes, networks have a generative mode and could be transformed into discriminative networks to deeper learning.

- **Supervised learning deep networks**

These networks are used when data target label are available, models that can directly provide power discriminant for classification purposes.

- **Hybrid deep networks**

These networks are the combination of the two types of networks mentioned above, so unsupervised deep networks could provide an excellent initialization on the basis of which discrimination could be examined.

Next figure summaries the Classification of deep learning architecture:

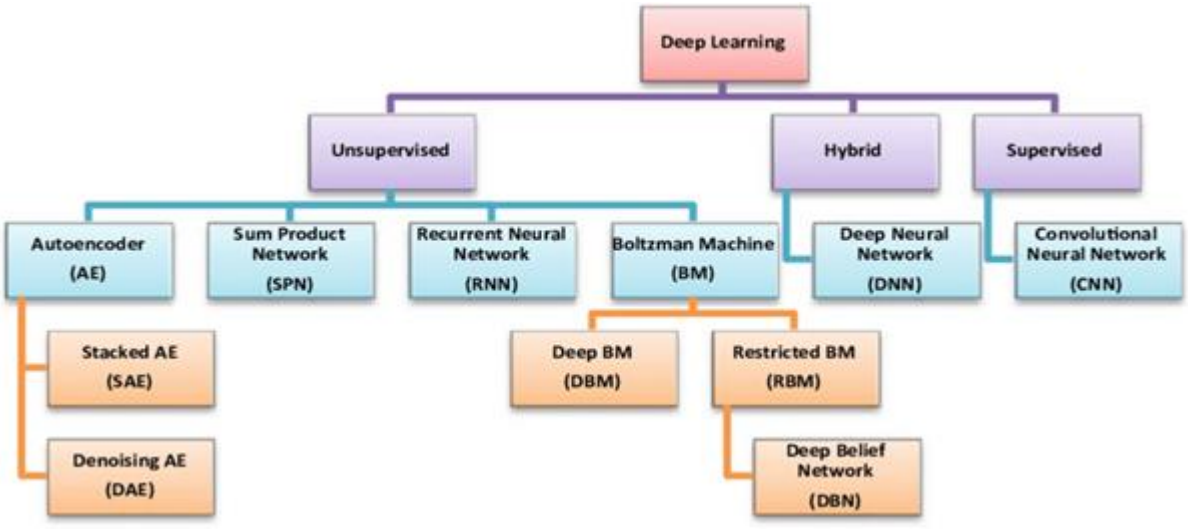



**Figure 7:Classification of deep learning architecture** [17]

## II.8.2 Deep Learning models

Practically all deep learning algorithms are based on neural networks, which share similar basic properties. These networks consist of interconnected neurons organized in layers. What distinguishes it is the specific structure of the network, that is, the way the neurons are arranged, and sometimes also the way they are trained we present the main deep learning methods, Although Figure  list is not exhaustive, it includes most of the algorithms currently commonly used:

DL is a set of prediction models that are based on Artificial Neural Networks (ANNs),an ANN is a generic term that encompasses any structure of interconnected neurons sending information to each other.

The key difference between Deep Neural Networks (DNNs) and the simpler single-hidden-layer neural networks is the large depth of the former; that is, the high number of hidden layers participating in the multistep process of pattern recognition. Specifically, a

DNN consists of an input layer, a suitable number (more than one) of hidden layers, and an output layer,each layer of a DNN is composed of neurons that are capable of producing nonlinear outputs from their inputs.

Data is propagated to the hidden layers by the input layer neurons that initially receive it. Then, the neurons in the hidden layers generate weighted sums of the input data on which they apply specific activation functions (e.g., ReLU or tanh).

| Algorithm | Learning type | Functions |
|---|---|---|
| AutoEncoder | Unsupervised learning | Feature extraction; feature reduction; denoising |
| Restricted Boltzmann machine (RBM) | Unsupervised learning | Feature extraction; feature reduction; denoising |
| Generative adversarial network (GAN) | Unsupervised learning | Data augmentation; adversarial training |
| Deep belief network (DBN) | Supervised learning | Feature extraction; classification |
| Deep neural network (DNN) | Supervised learning | Feature extraction; classification |
| Convolutional neural network (CNN) | Supervised learning | Feature extraction; classification |
| Recurrent neural network (RNN) | Supervised learning | Feature extraction; classification |

**Figure 8:Existing deep learning models [10]**

Subsequently, these outputs are propagated to the output layer, where the final results are presented[18], Figure 9(a) shows an example of a DNN.

### II.8.2.1 Convolutional Neural Networks (CNNs)

Convolutional neural networks (CNN) can be viewed as a regularized version of multilayer perceptrons with a strong inductive bias for learning translation-invariant image representations. There are four main components in CNNs: (a) learnable convolution filterbanks, (b) nonlinear activations, (c) spatial coarsening (via pooling or strided convolution), (d) a prediction module, often consisting of fully connected layers that operate on a global instance representation.

In CNNs we use convolution functions with multiple kernels or filters with trainable and shared weights or parameters, instead of general matrix multiplication. These filters/kernels are matrices with a relatively small number of rows and columns that convolve over the input to automatically extract high-level local features in the form of feature maps. The filters slide/convolve (element-wise multiply) across the input with a fixed number of strides to produce the feature map and the information thus learnt is passed to the hidden/fully connected layers. Depending on the input data, these filters can be one, two, or three-dimensional.

Similar to the fully connected NNs, nonlinearities such as ReLU are then applied that allows us to deal with nonlinear and complicated data. The pooling operation preserves spatial invariance, downsamples and reduces the dimension of each feature map obtained after convolution , these downsampling/ pooling operations can be of different types such as maximum-pooling, minimum-pooling, average pooling, and sum pooling. After one or more convolutional and pooling layers, the outputs are usually reduced to a one-dimensional global representation. CNNs are especially popular for image data[19].

### II.8.2.2 Recurrent Neural Network (RNN) and Long ShortTerm Memory (LSTM)

Fully connected neural networks and convolutional networks are types of feed-forward neural networks, meaning that signals move without looping within the networks, these networks are typically used for tasks with fixed-size inputs and outputs, but this configuration can hinder some tasks. Additionally, convolutional and fully connected networks do not take the order of inputs into account, which can cause problems for some tasks.

A recurrent neural network (RNN) is a type of network that has an internal state, or memory, based on the input data already provided. The output of an RNN is a combination of its internal state and the last input sample, and this internal state changes to incorporate the new data. RNNs are particularly suited to tasks dealing with sequential data, such as text or time series, thanks to their ability to memorize information from previous time steps.

Figure 9(b) illustrates a conventional RNN unfolded in time that can be trained across many times steps using backpropagation through time (BPTT). Despite their efficiency in modelling sequential data, RNNs suffer from the so-called vanishing gradient problem, that occurs when the output at any given time step depends on inputs much earlier in time ,LSTM architecture has been developed in order to address this issue, As shown in Figure 9 (e), each unit in the LSTM model has a cell memory with a state that stores information about the input sequences across time steps,the reading and modifying access to the memory units are controlled through sigmoid gates. LSTM models perform better than RNN models when data is characterized by a long dependency on time. Such long dependency can be observed in data generated by IoT networks or complex systems , Overall, the use of LSTM and its variants (e.g., Convolutional Long Short-Term Memory - ConvLSTM ) shows promise in improving the attack detection and prediction accuracy in settings where data is time-dependent[18].

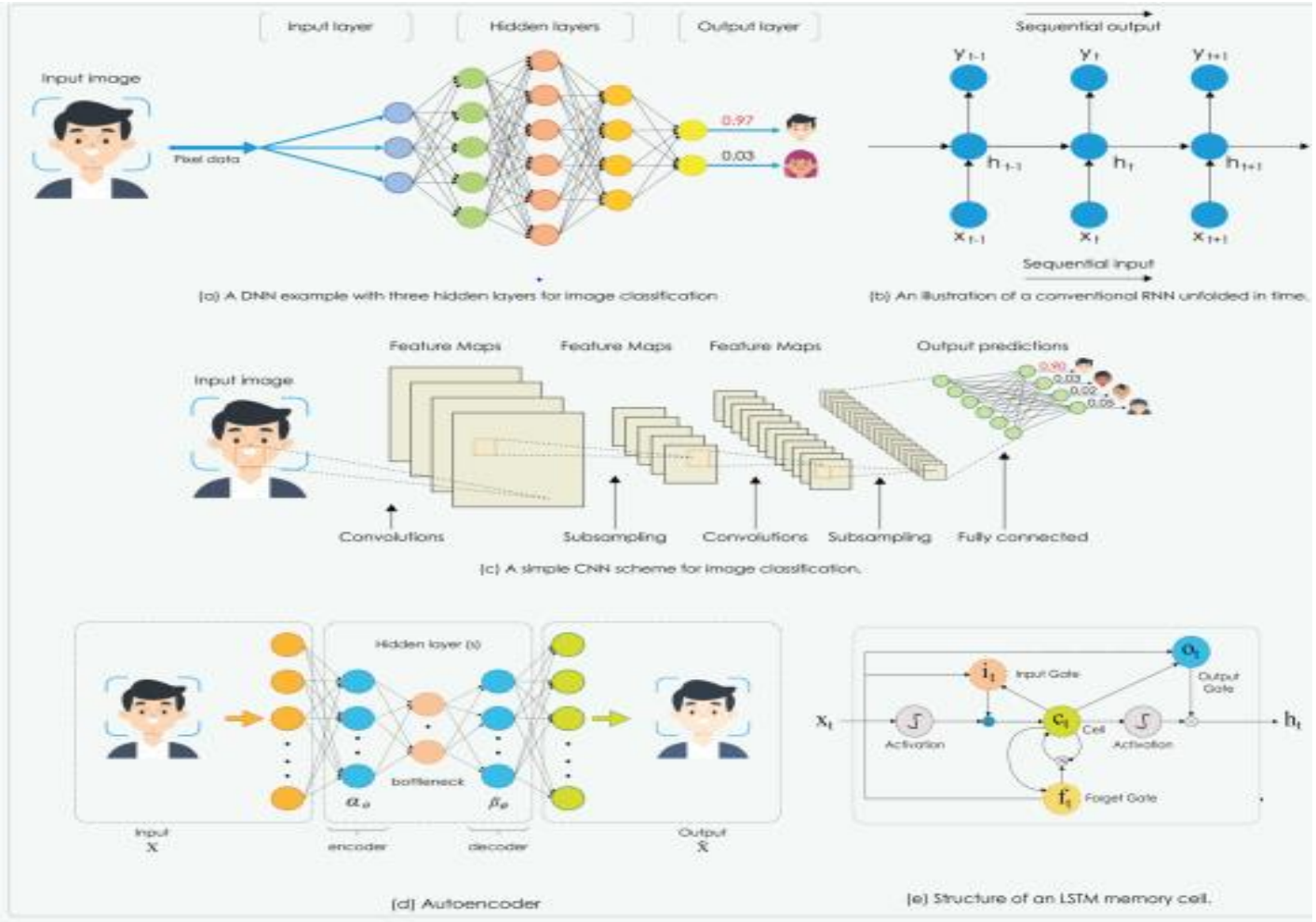



**Figure 9:DNN, RNN, CNN, and LSTM Architectures [5]**

## II.8.2.3Autoencoder (AE)

A class of unsupervised discriminative feature learning algorithms, in which the desired output is set to be equal to the input. As shown in Figure 9 (d), AEs generally consist of two parts, namely the encoder and the decoder. The encoder transforms the input data into a simpler version, reducing the dimensions, while the decoder reconstructs this data from this simplified version to find the original input. There are different types of AEs, such as sparse AEs, denoising AEs, stacked contractive AEs, adversarial AEs and variational autoencoders (VAEs) . Recently, the concept of autoencoder has become most widely used for network intrusion and spam detection tasks, due to the reconstruction from the input to the output layer.

### II.8.2.4 Restricted Boltzmann machine (RBM)

Boltzman's machines restricted neural networks are a special type of generative neural network, It consists of a visible layer and a hidden layer, nodes within one layer are not connected, but nodes in different layers are fully connected. RBMs are generally used for feature extraction or denoising.

### II.8.2.5 Deep belief network (DBN)

As can be seen in **Figure 10**(a) DBNs is a type of deep neural network that is essentially a probabilistic generative model including several layers of hidden variables. These networks have both directed and undirected edges ,a DBN is formed by connecting multiple RBM layers and are used in the case of feature extraction or classification.

### II.8.2.6 Generative adversarial network (GAN)

GAN consists of a generator that generates synthetic data similar to the real data, and a discriminator that distinguishes between the generated and the real data [20] , the training process of classical GANs is very sensitive to model structures, learning rates and other hyper-parameters. In the cybersecurity domain, GANs are typically applied to overcome the data imbalance problem by using artificial variations to minimize any bias in data collection [18]. An example of GAN is depicted in Figure 10 (b).

### II.8.2.7 Deep Reinforcement Learning (DRL)

Deep Reinforcement Learning (DRL) combines Reinforcement Learning (RL) and Deep Neural Networks (DNNs) to develop intelligent agents capable of optimizing action policies to maximize long-term rewards in complex and high-dimensional environments , RL focuses on finding the optimal policy of actions based on environmental states, while DNN is used to represent and approximate action values to estimate the quality of actions in different states. Integrating deep learning into traditional RL enables DRL to overcome challenges such as dynamic security, complexity and high dimensionality, particularly applying to Controlled Production Systems (CPS), cyber strategies -defense, and autonomous intrusion detection.

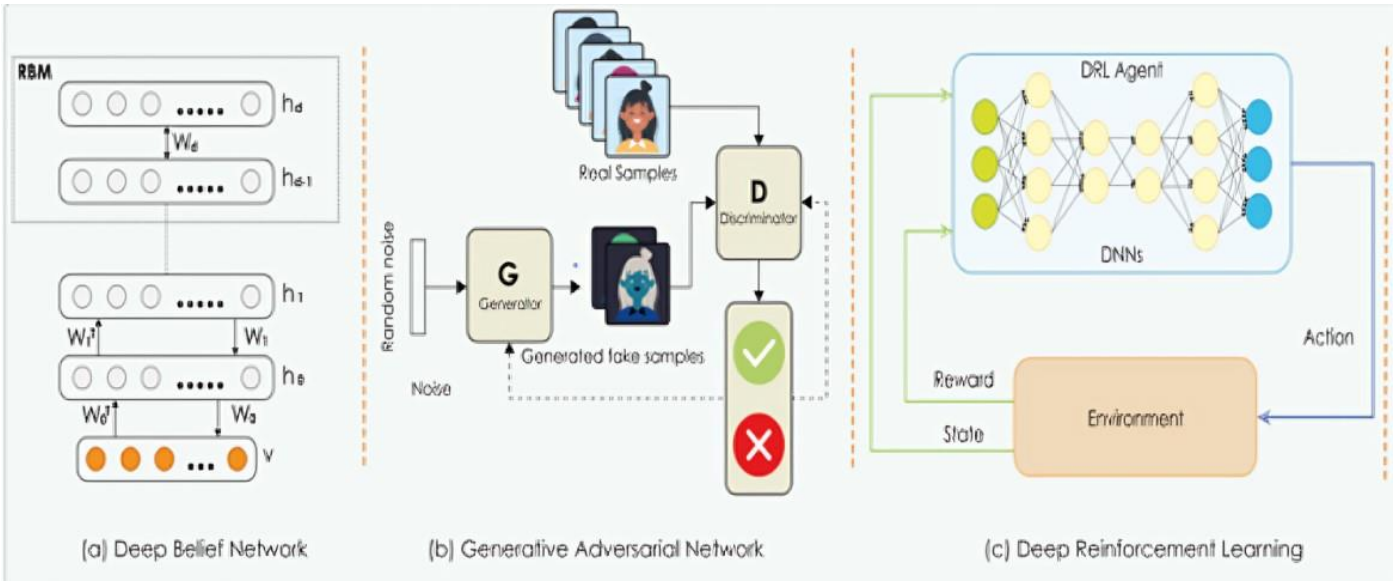



**Figure 10:** DBN, GAN, DRL architectures [5].

### II.8.2.8 Bidirectional Encoder Representations from Transformers (BERT)

Bidirectional Encoder Representations from Transformers (BERT) is a deep learning model and it is specifically a type of transformer model, which is a neural network architecture known for its effectiveness in handling sequential data such as natural language, it is a powerful pre-trained language model developed by Google. BERT is designed to understand the context of words in a sentence by looking at both the words before and after a target word, making it bidirectional. This capability allows BERT to capture the nuanced meanings of words based on their context, which is particularly useful for tasks such as hate speech detection where the meaning can change depending on the surrounding words[21].

BERT's architecture is based on Transformers, which are neural network models known for their ability to handle sequential data and capture long-range dependencies. The pre-training of BERT involves two tasks: masked language modeling (MLM) and next sentence prediction (NSP), which help it learn deep contextual representations of words[21].

## II.9 Cybersecurity Datasets

Benchmark datasets are crucial for cybersecurity researchers because they enable training, validation, and testing of deep learning-based security solutions. They are essential for reproducing experiments and comparing the performance of different methods using the same data. In this section, we present an overview of the most relevant datasets in the cybersecurity domain, grouped by specific application or category.

The most frequently used datasets for network intrusion detection include **KDD Cup 1999** and its evolution **NSL-KDD**. These sets contain traffic records classified into five categories: Normal, Probe, Remote to Local (R2L) Attacks, User to Root (U2R) Attacks, and Denial of Service (DoS) Attacks. It is important to note that this data is from the year 1998. **NSL-KDD** was developed to address redundancy and bias issues present in its predecessor, KDD Cup 1999.

**BoT-IoT** is a network of multiple bots designed to carry out malicious activities on a target network. Managed by a botmaster via a command and control protocol, it allows malicious attacks to be coordinated remotely. These botnets can vary in size, from small botnets with a few hundred bots to large botnets of up to 50,000 bots. Despite their discretion, these botnet viruses can remain effective and operational for years, distributed by hackers who operate without leaving a trace[22] , other notable datasets include **CTU-13,** which contains raw packet data (PCAP files) and examines 13 botnet attack scenarios.

**UNB ISCX 2012**, created by the Canadian Cybersecurity Institute at the University of New Brunswick, captures network traffic in a simulated environment over a 7-day period. AWID focuses on 802.11 networks, with a small network environment of 10 clients capturing WLAN traffic. The **CIC-IDS2017** dataset, also created by UNB, includes different types of user profiles and multi-level attacks. **CSE-CIC-IDS2018** contains 6 types of network attacks and was generated using synthetic user profiles.

**CIC-DDoS2019** dataset includes many different DDoS attacks carried out across application layer protocols using TCP/UDP. IoT-23, created in 2019, consists of 20 captures with malware activity and three captures of benign IoT traffic. TON_IoT, generated in 2019 by Abdullah et al., includes various types of IoT data and IoT traffic collected from a medium-sized network at the Cyber Range and IoT Labs at UNSW Canberra, Australia. And also the LITNET-2020 dataset which contains feature vectors generated during 12 attacks on general purpose computers deployed on an academic network, collected over a period of 10 months[18].

**CICIOT 2023** dataset which was created by the Canadian Institute for Cyber Security (UNB) and contains different types of user profiles generating background traffic, as well as multi-level attacks such as Heartbleed and DDoS.

Figure 11 shows Diagram of the shared and exclusive features of four IDS datasets:

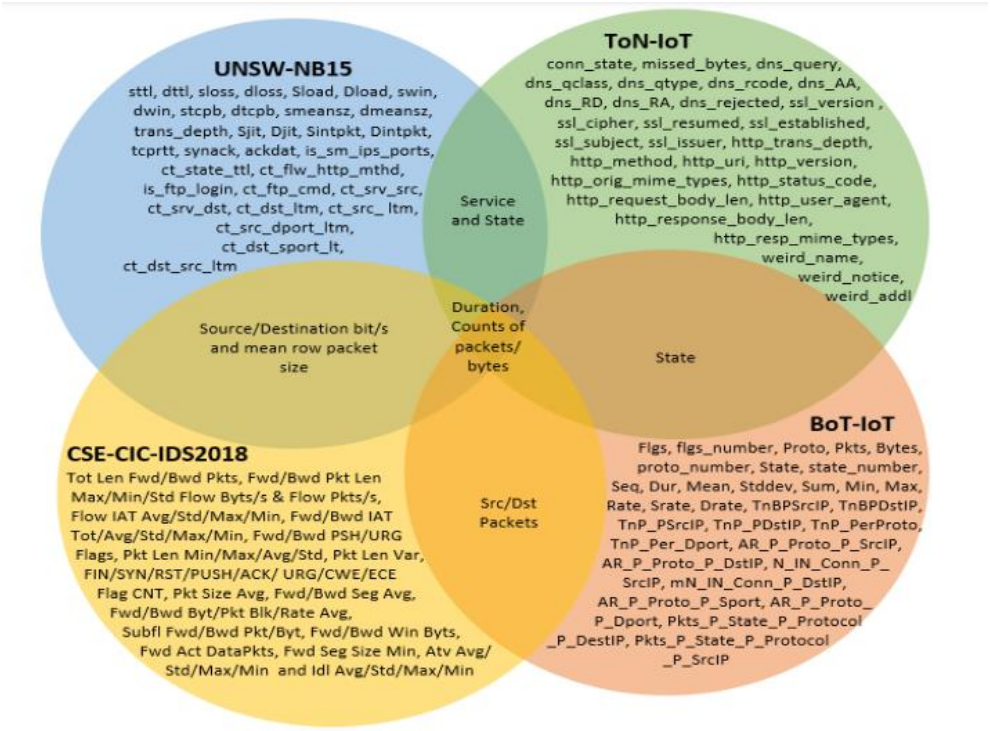



**Figure 11:Diagram of the shared and exclusive features of four IDS datasets** [18]

## II.10 Related Works

With Internet usage exponentially increasing and hacking on the rise, many researchers have turned to deep learning to improve intrusion detection in IoT environments, this trend aims to more effectively identify new forms of complex cyberattacks,the table below compares several deep learning-based intrusion detection frameworks, highlighting the different strategies, methodologies and associated performance.

Methods adopted in these frameworks include the use of convolutional neural networks CNN for analyzing spatial data, recurrent neural networks (RNN) for processing sequential data, and attention mechanisms for focusing on the relevant parts of the data. For example, a combination of CNN and LSTM (Long Short-Term Memory) has been implemented to extract spatial and temporal features from network traffic data. To optimize detection, advanced feature optimization techniques are integrated, such as the application of genetic algorithms and autoencoders.

The datasets used in these studies include recognized benchmarks like CIC-IDS2017 and CIC-DDOS 2019, providing a diverse range of attack scenarios to evaluate the performance of the models. Model performance is improved using metrics such as precision and recall, helping to

measure the effectiveness of intrusion detection. Additionally, model optimization techniques, such as automatic feature selection, are explored to improve the transparency and

| **Article Title** | **Journal** | **Date** | **Algorithms Used** | **Results** | **Notes** | **Dataset Used** |
|---|---|---|---|---|---|---|
| DL-IDS: Extracting Features Using CNN-LSTM Hybrid Network for Intrusion Detection System [23] | Hindawi | August 28, 2020 | CNN, LSTM | Accuracy: 98.67%, F1-Score: 93.32% | Used category weight optimization, converted data packets into traffic images, six-step data processing method Multiclass classification | CICIDS2017 |
| Anomaly-Based Intrusion Detection System for DDoS Attack with Deep Learning Techniques | ScitePress | 2023 | DNN-based IDS, CNN, RNN | Accuracy: 99.80% (CSE-CIC-IDS-2018), 99.95% (CIC-DDoS-2019) | Applied label encoding, normalized numerical values, used PCA for dimension reduction, high accuracy in binary classification | CSE-CIC-IDS-2018, CIC-DDoS-2019 |

| | | | | | | |
|---|---|---|---|---|---|---|
| [24] | | | | | | |

interpretability of detection results.

These efforts aim to strengthen the security of IoT systems against a multitude of attacks, including denial of service (DoS) attacks, user-to-root (U2R) attacks, and probe attacks, by combining innovative approaches and optimization techniques, these deep learning-based intrusion detection frameworks help combat hacking and protect IoT networks from emerging threats, Machine learning and deep learning based methods have been widely used in recent studies on intrusion detection. With the developments in machine learning algorithms and deep learning algorithms, thanks to access to big data, more effective intrusion detection systems are being developed in this area. We have presented some of the current studies on attacks in systems in the literature in this section.

**Table 3:**Related works.

| | | | | | | |
|---|---|---|---|---|---|---|
| Anomaly-Based Intrusion Detection System for IoT Networks through Deep Learning Model[25] | Researchgate | 2022 | CNN 1D GCNN | Accuracy : 99.68% for binary classes, 99.90% for 8-classes, and 99.95% for 13-classes Precision: 95.55% - 99.51% | -binary and multi class classification | CicDDOS 2019 |
| A Content-Based Deep Intrusion Detection System[26] | Springer | 2022 | LSTM | Precision: 99.2% (CIC-IDS2017), 93.3% (CSE-CIC-IDS2018), Recall: 99.8% (CIC-IDS2017), 92.3% (CSE-CIC-IDS2018) | Detected content-based attacks (SQL injections, XSS, viruses), novelty in using raw content bytes enriched with flows Multiclass classification | CIC-IDS2017, CSE-CIC-IDS2018 |
| Intrusion Detection System for Internet of Vehicles Based on Ensemble Learning and CNN[27] | Journal of Physics: Conference Series | 2022 | CNN Ensemble Learning, VGG16, MobileNetV2 DenseNet | Accuracy: 100%, F1-Score: 100%, Detection Time: 1.0 ms to 2.8 ms | The proposed IDS model combines CNNs with ensemble learning. Utilizes VGG16, MobileNet V2, and DenseNet as base learners. Ensemble strategy bagging is applied to improve accuracy while keeping the detection time short. The model is validated on the Car-Hacking Dataset, demonstrating high effectiveness. | Car-Hacking Dataset |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | | | Multiclass classification | |
| A Survey and Analysis of Intrusion Detection Models[28] | Journal of Big Data | 2020 | Feed-forward Neural Networks, Autoencoders, Deep Belief Networks, LSTMs, Rnn | High accuracy and precision for deep learning models | Every group worked on a combination of models Evaluated on CICIDS2017 and CICIDS2018 datasets. Mlticlass classification | CICIDS2017, CICIDS2018 |
| Conceptualisation of Cyberattack Prediction with Deep Learning[29] | Springer | 2020 | -Expectation Maximisation algorithm -Various Deep Learning Models (dffnn,dnn, pca) | Accuracy: over 99% for most attack classes | Focuses on predicting various attack classes, demonstrating the effectiveness of deep learning in IDS. Multiclass | CICIDS2017 UNSW_NB15 |
| A Review of DDoS Evaluation Dataset: CICDDoS2019 Dataset [30] | Future Internet | | Deep Learning Models | Accuracy: 99.1% Precision: 98.9%, Recall: 99.0%, F1-Score: 98.95% | Reviews the CICDDoS2019 dataset, explores various deep learning models for detecting DDoS attacks. Provides insights into the dataset's features and model performance. Mentioned hyperparameters: learning rate, batch size,Binary classification Oversampling for balancing the data | CICDDoS2019 |

| | | | | | | |
|---|---|---|---|---|---|---|
| DL-IDS: a deep learning–based intrusion detection framework for securing IoT[15] | Transactions on Emerging Telecommunications Technologies. | October 2021 | (SMO) and (SDPN) | The higher accuracy is achieved on the layer 3 with the lambda value equals to (0.001) Accuracy (%) 99.02 Precision (%) 99.38 Recall (%) 98.29 F1-Score (%)98.83 | SMO for Feature Selection. SDPN Classification. The methodology integrates SMO for feature selection in each layer of SDPN, optimizing feature learning and enhancing performance. The integration of SMO and SDPN in their methodology enhances the efficiency and effectiveness of the IDS by optimizing feature selection and reducing training time. | NSL-KDD benchmark dataset{ |
| Attentive transformer deep learning algorithm for intrusion detection on IoT systems using automatic Xplainable feature selection [31] | PLOS ONE | October 2023 | decision trees, random forests, XGBoost, LightGBM, and Extra Tree classifiers. deep learning models CNN DNN, RNN attentive mechanisms (to | CIC-IDS2017 accuracy 97%. CSE-CICIDS2018 accuracy 95%. CIC-DDoS2019 accuracy 98%. | A wide range of models was used, This demonstrates a comprehensive approach to finding the best-performing model for IDS. | CIC-IDS2017 CSE-CICIDS2018 CIC-DDoS2019 |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | automatically select salient features). | | | |
| A hybrid deep learning-based intrusion detection system for IoT networks[5] | aimspress | 2023 | RNN - GRU SMO -SDPN | Network Flow Datasets: Achieved an accuracy of 99%. Application Layer Datasets: Achieved an accuracy of 98% | The proposed model was validated through three distinct experiments, demonstrating its robustness and high performance. | ToN-IoT |
| Sequential Model Based Intrusion Detection System for IoT Servers Using Deep Learning Methods[32] | mdpi | 2021 | Text-CNN with GRU SVM-RBMS | precision : 0.97. | Text-CNN uses the backpropagation algorithm for training the network, with Stochastic Gradient Descent (SGD) employed to update the gradient during training. | KDD99 ADFA-LD |
| Enhancing IoT Security with CNN and LSTM-Based Intrusion Detection Systems[33] | IEEE. | 2024 | CNN and LSTM | accuracy:98.42% f1score:98.57% for normal class precision is 90% recall is 61% and f1score is 71% accuracy 90% for normal class and accuracy of 99% for the attacks class | Features are extracted and organized into a matrix of vectors, transformed into a two-dimensional format for model input, and labels are converted to binary (benign or malicious). | CIC-IoT2023 |

Several studies have explored the application of deep learning techniques to Intrusion Detection Systems (IDS), highlighting diverse methodologies and achieving notable results.

Sun et al [23] proposed a hybrid CNN-LSTM model, converting data packets into traffic images through a six-step data processing method. This approach achieved an accuracy of 98.67% and an F1-score of 93.32% on the CICIDS2017 dataset, emphasizing the efficacy of combining convolutional and recurrent neural networks for feature extraction and classification in network traffic data.Smith et al. (2023) developed an IDS using DNN, CNN, and RNN models, applying label encoding and PCA for dimensionality reduction. The system attained accuracies of 99.80% and 99.95% on the CSE-CIC-IDS-2018 and CIC-DDoS-2019 datasets, respectively, demonstrating high performance in both binary and multiclass classification scenarios.

T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj [25] utilized CNN 1D and GCNN models for detecting anomalies in IoT networks. Their study achieved accuracies of 99.68% for binary classes and up to 99.95% for 13-class scenarios, with precision ranging from 95.55% to 99.51%. The study, validated on the CICDDoS 2019 dataset, showcased the models' effectiveness in handling diverse IoT network threats.

M. Soltani, M. J. Siavoshani, and A. H. Jahangir [26] presented an LSTM-based IDS focusing on content-based attacks such as SQL injections and XSS. The system achieved high precision (99.2% for CIC-IDS2017 and 93.3% for CSE-CIC-IDS2018) and recall rates (99.8% and 92.3%, respectively), utilizing raw content bytes enriched with flow information to detect sophisticated network attacks.

A. Luo [27] combined CNNs with ensemble learning techniques using VGG16, MobileNet V2, and DenseNet. The proposed model achieved an accuracy and F1-score of 100%, with detection times ranging from 1.0 to 2.8 milliseconds, validated on the Car-Hacking Dataset. This work demonstrated the potential of ensemble methods to enhance detection speed and accuracy in vehicular networks.

J. L. Leevy and T. M. Khoshgoftaar [28] conducted a comprehensive survey examining various deep learning models, including feed-forward neural networks, autoencoders, deep belief networks, LSTMs, and RNNs. Evaluated on the CICIDS2017 and CICIDS2018 datasets, the study reported high accuracy and precision in multiclass classification tasks, providing valuable insights into the strengths and limitations of different IDS approaches.

A. E. Ibor, F. A. Oladeji, O. B. Okunoye, and O. O. Ekabua [29] focused on predicting cyberattacks using a combination of expectation maximization algorithms and various deep learning models, including DFFNN and PCA. The study achieved over 99% accuracy for most attack classes on the CICIDS2017 and UNSW_NB15 datasets, demonstrating the effectiveness of deep learning in cyberattack prediction.

M. C. P. Saheb, M. S. Yadav, S. Babu, J. J. Pujari, and J. B. Maddala [30] reviewed the CICDDoS2019 dataset, exploring multiple deep learning models for DDoS attack detection. Achieving an accuracy of 99.1%, precision of 98.9%, recall of 99.0%, and F1-score of 98.95%, the study highlighted the dataset's features and model performance, emphasizing the importance of hyperparameters like learning rate and batch size in model optimization.

B. Noor and S. Qadir [1] combined the Spider Monkey Optimization algorithm (SMO) with a Stacked-Deep Polynomial Network (SDPN), achieving an accuracy of 99.02%, precision of 99.38%, recall of 98.29%, and F1-score of 98.83%. The study demonstrated the integration of SMO for feature selection in each layer of SDPN, optimizing feature learning and enhancing performance on the NSL-KDD benchmark dataset.

D. Zegarra Rodríguez, O. Daniel Okey, S. S. Maidin, E. Umoren Udo, and J. H. Kleinschmidt, [31] discussed the development of IDS using a combination of tree-based algorithms and deep learning models with attentive mechanisms for automatic feature selection. The study reported accuracies of 97% on CIC-IDS2017, 95% on CSE-CICIDS2018, and 98% on CIC-DDoS2019, showcasing a comprehensive approach to identifying the best-performing models for IDS.

N. W. Khan et al., [5]proposed a hybrid model combining RNN-GRU with SMO and SDPN, achieving 99% accuracy on network flow datasets and 98% on application layer datasets. Validated through three distinct experiments, this study demonstrated the robustness and high performance of the model on the ToN-IoT dataset.

M. Zhong, Y. Zhou, and G. Chen [32] employed a Text-CNN with GRU-SVM-RBM approach, achieving a precision of 0.97. The Text-CNN model utilized backpropagation for training and stochastic gradient descent (SGD) for gradient updates, validated on the KDD99 and ADFA-LD datasets, highlighting the effectiveness of sequential models in detecting intrusions on IoT servers.

A. Gueriani, H. Kheddar, and A. C. Mazari [33] present a CNN-LSTM hybrid model for enhancing IoT security through effective intrusion detection. The model is trained and tested

on the CIC-IoT2023 dataset, achieving high accuracy (98.42%) and an F1-score of 98.57%. The results demonstrate the model's robustness and effectiveness in detecting and classifying network intrusions.

These studies collectively underscore the advancements and diversity in applying deep learning techniques to IDS, each contributing unique methodologies and results that drive the field forward.

In this section, IDS system designs in the literature are reviewed,when we look at the literature studies, we see that many different machine and deep learning methods are used to detect different types of attacks,when analyzing the performance of these methods, it was determined that they have advantages and disadvantages compared to each other. In addition to system designs using a single algorithm, it has been found that hybrid approaches are frequently preferred in designs.

# III Chapter III

# Design of the Models Used

# III.1 Introduction

In this chapter, we detail the conception of our innovative method intended to reduce dimensionality and classify high-dimensional voluminous data, coming from the network flow, in order to identify intrusions.

Next figure describes the process of creation IDS using Deep learning:

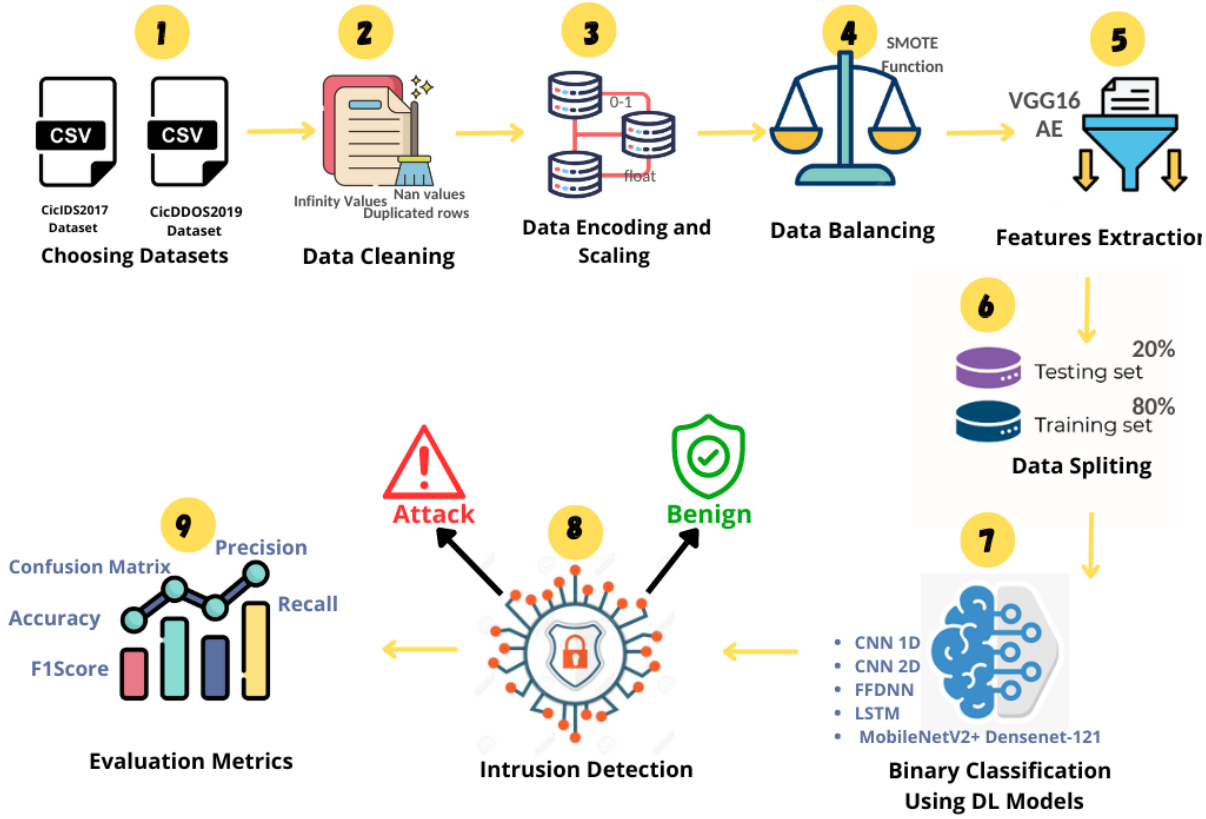



**Figure 12:Plan of creation IDS using Deep learning**

1. Choosing Datasets : We chose CicIDS2017[1] and CicDDOS 2019[2] datasets to train our Models because of their important features and due to the existing related works that used these datasets in order to compare results of our models with their results.

[1] https://www.unb.ca/cic/datasets/ids-2017.html
[2] https://www.kaggle.com/datasets/rodrigorosasilva/cic-ddos2019-30gb-full-dataset-csv-files

2. Data Cleaning: Remove rows with missing values, Drop infinite and Nan Values, Drop duplicated rows .
3. Data Encoding and Scaling: encoding categorical columns, Converting columns to float and scale all the values between 0 and 1 in order to can use them with Deep Learning Models.
4. Data Balancing:Both of datasets are imbalanced , it is important to balance the dataset Otherwise the results will be favorised for one class than the second , for that we used SMOTE function which works to oversample the miniority class.
5. Features Extraction: we tried 2 experiments, the first we used Autoencoder , for the second we used VGG16 (after we had transformed the tabular dataset to image) .
6. Data Spliting: we used 80% of the data training and 20% for testing.
7. Binary Classification Using DL Models: we used CNN-1D , CNN-2D, FFDNN , LSTM, MobileNetV2 and Densenet-121 models for binary classification.
8. Intrusion Detection: As result of the model , it detect if there's an intrusion or not .
9. Evaluation Metrics : Accuracy , Recall , F1score , Precision , Training and Testing Accuracy Graph , Confusion Matrix are used to evaluate the performance of each model.

Next table resumes the Steps for Intrusion Detection System Conception:

**Table 4:**Steps for the Intrusion Detection System Conception

| Steps | Description |
|---|---|
| **Normalization of the Dataset** | • Encoding categorical columns<br>• removing rows with missing values<br>• converting columns to float and handling infinite values. |
| **Data balancing** | • Using Oversampling function SMOTE |
| **Feature extraction** | • Selection and extraction of features using a deep autoencoder. |

| | |
|---|---|
| **Classification** | • Classification using a deep learning models: CNN 1D ,CNN 2D , FFDNN , LSTM, Combination of MobileNetV2+ DenseNet 121<br>• Binary Classification |
| **Evaluation of the models** | • Accuracy, Precision, Recall, F1score<br>• confusion matrix<br>• The train and validation accuracy plot |

## III.2 Preprocessing and Data Analysis

The work carried out in this study is based on two references datasets for intrusion detection known as CICIDS-2017 and CICDDOS2019; this because they contain various features and meet the majority of criteria. The datasets used are beneficial because they handle most real network attacks. Additionally, it contains most attacks on data networks,these datasets used in the research were developed by the University of New Brunswick (UNB) in collaboration with the Canadian Cyber Security Institute [34].

Canadian Institute for Cybersecurity (CIC) at the University of New Brunswick (UNB) initiated the creation of the CICDDoS 2019 dataset. This dataset contains 50,063,112 instances including 50,006,249 DDoS attacks and 56,863 benign (normal) instances), a compilation of 86 network traffic packet attributes generated by an open-source tool. The latter produces network packets and collects attributes from them. The CICDDoS2019 dataset, accessible through the University of New Brunswick, brings together representative examples of common DDoS attacks and legitimate network traffic, providing an accurate picture of real-world data. It integrates packet captures (PCAPs) and network traffic analyzes performed with CICFlowMeter-V3, providing labeled streams including timestamps, source and destination IP addresses, source and destination ports, protocols and attack types (in CSV files)[8],intended to promote cybersecurity research, this dataset helps with the development and evaluation of intrusion detection systems (IDS) and DDoS attack prevention measures.

## III.3 CICIDS2017 Dataset Description

CIC-IDS2017 [3]dataset, developed by the Canadian Cybersecurity Institute (CIC) in collaboration with the University of New Brunswick (UNB), is a benchmark intrusion dataset known for its diversity of characteristics and its compliance with most network security criteria. This dataset, captured between July 3 and 7, 2017, includes both normal and abnormal traffic, with normal activity reported only on Mondays. The data is tabular dataset covers a five-day period, was collected using a B-Profile system and analyzed with **CICFlowMeter,** revealing streams labeled based on their timestamps, destination ports, sources, addresses Destination IP, protocols and attack types,the abstract behaviors of 25 users were identified according to SSH, HTTP, FTP and HTTPS. The dataset contains 2,830,743 network traffic samples, with 80.30% benign traffic and 19.70% attack traffic, including 14 common attack categories such as DoS, DDoS, Botnet, PortScan, and attacks. Web , Data was extracted for 79 network traffic characteristics, with the last column indicating the multiclass category. This dataset was designed to meet eleven performance evaluation criteria, outperforming publicly available datasets from 1998 to 2016 [35] ,it is particularly relevant to cybersecurity research, providing an overview of network attacks from real world and providing a framework for the evaluate on and development of intrusion detection systems.

Figure 13 shows a screenshot of part of the CICIDS 2017 dataset:

| | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Destination | Flow Duratio | Total Fwd Pa | Total Backw | Total Length | Total Length | Fwd Packet | Fwd Packet | Fwd Packet | Fwd Packet | Bwd Packet L | Bwd Packet | Bwd Packet | Bwd Packet | Flow Bytes/s | Flow Packet | Flo |
| 2 | 54865 | 3 | 2 | 0 | 12 | 0 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 4000000 | 666666.6667 | |
| 3 | 55054 | 109 | 1 | 1 | 6 | 6 | 6 | 6 | 6 | 0 | 6 | 6 | 6 | 0 | 110091.7431 | 18348.62385 | |
| 4 | 55055 | 52 | 1 | 1 | 6 | 6 | 6 | 6 | 6 | 0 | 6 | 6 | 6 | 0 | 230769.2308 | 38461.53846 | |
| 5 | 46236 | 34 | 1 | 1 | 6 | 6 | 6 | 6 | 6 | 0 | 6 | 6 | 6 | 0 | 352941.1765 | 58823.52941 | |
| 6 | 54863 | 3 | 2 | 0 | 12 | 0 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 4000000 | 666666.6667 | |
| 7 | 54871 | 1022 | 2 | 0 | 12 | 0 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 11741.68297 | 1956.947162 | |
| 8 | 54925 | 4 | 2 | 0 | 12 | 0 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 3000000 | 500000 | |
| 9 | 54925 | 42 | 1 | 1 | 6 | 6 | 6 | 6 | 6 | 0 | 6 | 6 | 6 | 0 | 285714.2857 | 47619.04762 | |
| 10 | 9282 | 4 | 2 | 0 | 12 | 0 | 6 | 6 | 6 | 0 | 0 | 0 | 0 | 0 | 3000000 | 500000 | |
| 11 | 55153 | 4 | 2 | 0 | 37 | 0 | 31 | 6 | 18.5 | 17.67766953 | 0 | 0 | 0 | 0 | 9250000 | 500000 | |
| 12 | 55143 | 3 | 2 | 0 | 37 | 0 | 31 | 6 | 18.5 | 17.67766953 | 0 | 0 | 0 | 0 | 12300000 | 666666.6667 | |
| 13 | 55144 | 1 | 2 | 0 | 37 | 0 | 31 | 6 | 18.5 | 17.67766953 | 0 | 0 | 0 | 0 | 37000000 | 2000000 | |

**Figure 13: Screenshot of part of CICIDS 2017 dataset**

## III.4 CICDDDS2019 Dataset Description

CICDDoS2019 [4]dataset developed by the Canadian Institute for Cybersecurity (CIC), is designed to facilitate research in the area of network intrusion detection systems (NIDS) and specifically targets Distributed Denial of Service (DDoS) attack scenarios ,the dataset is tabular

[3] https://www.unb.ca/cic/datasets/ids-2017.html

[4] https://www.unb.ca/cic/datasets/ddos-2019.html

dataset like CicIDS2017 , it includes benign and DDoS attack traffic, offering a comprehensive view of various attack vectors and patterns that are pertinent to real-world network security challenges.

**Key Features of the CICDDoS2019 Dataset:**

- **Diverse Attack Types:** The dataset encompasses a wide range of DDoS attacks, including but not limited to UDP Flood, TCP SYN Flood, HTTP Flood, and ICMP Flood. Each attack type is represented with multiple instances to ensure variability and realism.
- **Traffic Capture:** Traffic was captured using CIC's own network traffic generator, capturing detailed packet-level data. This includes features such as source and destination IP addresses, source and destination ports, and various other packet attributes that are critical for identifying and analyzing DDoS attacks.
- **Temporal Dynamics:** The dataset is timestamped, allowing for the study of temporal patterns and the progression of DDoS attacks over time. This is particularly useful for developing time-series-based intrusion detection methods.
- **Feature Richness:** The CICDDoS2019 dataset includes a comprehensive set of features extracted from the raw packet data. These features are essential for machine learning and deep learning models and include statistical, network flow, and time-based attributes.
- **Imbalance:** The dataset exhibits a significant imbalance between attack and benign samples, reflecting real-world network traffic conditions where benign traffic far outweighs attack traffic. This poses a challenge for model training and necessitates the use of techniques such as data augmentation, resampling, or cost-sensitive learning.

The CICDDoS2019 dataset is widely used in research for developing and benchmarking NIDS. It provides a realistic and challenging environment for testing the effectiveness of various detection algorithms, particularly those based on machine learning and deep learning. Researchers leverage this dataset to evaluate the robustness, accuracy, and efficiency of their proposed methods in identifying and mitigating DDoS attacks.

## III.5 CICFlowMeter

CICFlowMeter, developed by the Intelligence and Control Center (CIC), is a network traffic analysis tool. This is the same entity that initiated the creation of the CICIDS2017 dataset. This tool can process either PCAP files or perform real-time analysis from a network adapter. After analyzing network packets, CICFlowMeter transforms them into streams and generates a CSV file containing 79 characteristics. This transformation capability is essential for our project, as it will allow training and testing of our model, as well as preprocessing real network traffic.

A notable element of CICFlowMeter is that it does not produce a CSV file directly corresponding to individual network packets, but rather a representation of the packets as a network stream. Unlike network packets, which are unidirectional segments of data, network flows organize these packets into a sequence, providing a better understanding of the communication flow between two endpoints. This feature makes CICFlowMeter particularly suitable for intrusion detection.

Regarding the CICIDS2017 dataset, it already includes CSV files generated by CICFlowMeter for each day of observed network traffic ,Despite the lack of direct use of CICFlowMeter in our process, the resulting CSV files had encoding and formatting issues, requiring a cleanup step to ensure their proper use[36].

## III.6 Data Preprocessing

Data Pre-processing is a technique used in data mining for the transformation of raw data into a format that can be utilized efficiently, it encompasses different techniques for data cleaning, data transformation, and data balancing, data cleaning is an essential step in pre-processing that removes the irrelevant and missing information from the dataset that could lead to the incorrect estimation of the target class there were a total of 8 different files that were merged into one single file and the entire work presented was relying on this file.

Different types of attacks noticed are displayed in Figure 14 ,in total, they were 2,830,743 examples included in 15 classes[33] .

| Attack Type | Description | Total Packet |
|---|---|---|
| Normal | Normal connection | 11,926,723 |
| FTP-Patator | File transfer protocol—brute force attack | 110,736 |
| SSH-Patator | Secure shell protocol—brute force attack | 136,073 |
| DoS Slowloris | Attackers flood the victim machine with malicious requests to overload victim systems | 47,596 |
| DoS Slowhttptest | Attackers flood the victim machine with malicious requests to overload victim systems | 39,254 |
| DoS Hulk | Attackers flood the victim machine with malicious requests to overload victim systems | 2,245,526 |
| DoS GoldenEye | Attackers flood the victim machine with malicious requests to overload victim systems | 106,177 |
| Heartbleed Port 444 | Exploited by sending a malformed heartbeat request with a small payload and large length field to the vulnerable party | 47,551 |
| Botnet ARES | Zombie machine controlled by bot onwer, can be used to perform various attacks | 9871 |
| DDoS LOIT | Distributed Denial of Service is an attempt to make victim services down by using multiple sources. This can be done by using botnet | 1,280,602 |
| Port Scan | Specify which port is opening for a particular service. Attacker use this to get information. | 327,253 |

**Figure 14:Description of CICIDS2017 Dataset [26]**

The normalization of data is important method in the intrusion detection. the methods for the most common uses include the z-score normalization and the min-max normalization, the normalization of data can take into account advanced numbers, so that the improvement of the models' convergence, the improvement of the precision of the model and the prevention of the expansion of the model's gradients in deep learning.

To apply deep learning models to these datasets, we need to convert them into a format suitable for deep learning, data preprocessing plays a crucial role.

After normalization, it is essential to verify the balance of the data to ensure the model's performance is not skewed by class imbalances, normalization standardizes the feature values, allowing the model to learn more effectively, but it does not address any underlying imbalances in class distribution, if the dataset remains imbalanced, the model may still favor the majority class, leading to biased predictions and poor performance on the minority class. Therefore,

after normalization, it's crucial to assess and, if necessary, apply balancing techniques such as oversampling or undersampling.

This step ensures that each class is adequately represented during training, promoting fair and accurate model predictions across all classes , table5 defines the steps of normalization and balancing :

**Table 5:**Normalization and Balancing Steps

| **Steps** | **Description** |
|---|---|
| **Create ''Class'' column** | Create Class column that takes "Benign" value or "Attack" depending to "Label" column value on order to use binary classification |
| **Encoding Categorical Columns** | Transform categorical columns ("Label" and "Class") into numeric format using LabelEncoder from sklearn.preprocessing. |
| **Dropping Rows with Missing infinity and Nan values** | Eliminate rows with missing ,infinity and Nan values using pandas dropna() method. |
| **Converting Columns to Float** | Convert all DataFrame columns to the decimal (float) data type |
| **Scaling the Data** | scale data between 0 and 1 using MinMaxScaler. |
| **Balancing dataset** | **Oversampling**: Increase the number of samples in the minority class by duplicating existing samples or generating new ones like SMOTE - Synthetic Minority Over-sampling Technique. |

## III.7 Proposed method for columns dimensionality reduction

Instance classification faces challenges, particularly due to the large number of attributes that define instances in the database. To improve the classification accuracy and reduce the calculation time, several dimension reduction techniques have been developed,as part of our study, we proposed a reduction of the dimensionality of the data using an auto-encoder type deep neural network.

An autoencoder is a type of neural network and unsupervised learning technique, Figure 15 shows its architecture, it is widely employed to reduce the feature set of a dataset to a stage that does not compromise overall classification performance. which consists of three types of layers:

1. The input layer 2. Hidden layers 3. The output layer

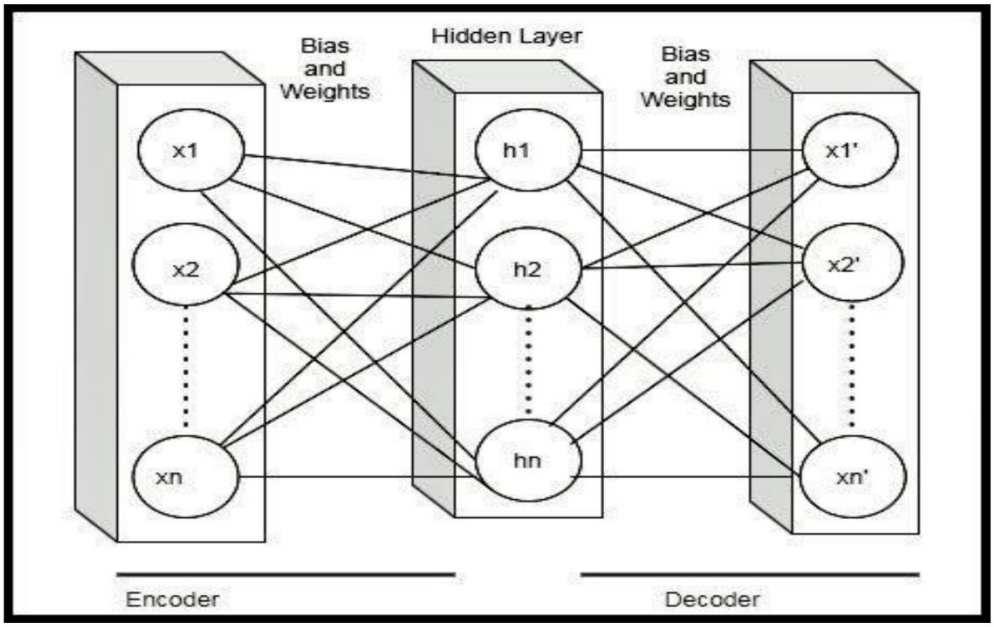



**Figure 15:Autoencoder architecture[27].**

After training, it can try to copy the input to the output. In other words, it can make the output the same as the input. The network can be considered as consisting of two parts: an encoder and a decoder. Simply put, while it is theoretically possible to create an autoencoder, it is not commonly done. To prevent the autoencoder from replicating perfectly, it is designed with specific constraints. These constraints limit its ability to replicate exactly, allowing it to learn relevant features of the data rather than simply replicating the inputs[37].

The data reconstruction process of AE is to try to learn the hidden features of the input data, and then try to reconstruct the data that is close to the original data according to the learned hidden features. In simple terms, the data reconstructed by the AE model can have maximum similarity with the original data. It can also show that the reconstructed dataset can reflect the importance of each feature in the original dataset to a certain extent. In other words,

reconstructed data based on hidden features can reflect the contribution of features in the original dataset to the reconstruction process.

As shown in Figure 15 , *X* is the input layer of AEs. *H* is the hidden layer of AEs. *X˜* is the output layer of AEs. *f* is the encoder and g is the decoder. The encoding and decoding processes are shown in (1), (2), respectively.

$$f : x \rightarrow h : hj = \sigma\left(W_{ei}x_i + b_{ei}\right) \quad (1)$$

$$g : h \rightarrow \tilde{x} : \tilde{x}_i = \sigma\left(W_{dj} + b_{dj}\right) \quad (2)$$

*xi* is the original feature. *xi* is converted to *hj* after being encoded by the encoder. *hj* is converted to *x˜i* after being decoded by the decoder g. *We* and *be* are the weight and bias of the encoder of *f* , respectively. *Wd* and *bd* are the weight and bias of the decoder g, respectively. σ is the activation function. σ can make the original features nonlinear, thereby improving the model's ability. Therefore, we can make the dimension of *H* smaller than *X*. When the AEs reconstruct the output feature *x˜*, which is as same as the input feature *x*, the hidden layer can capture the most prominent feature *h* among the original features at sparse AEs [38].
The learning process of the AEs can be described as minimizing the loss function. Normally, the loss function is the mean square error as follows.

$$MSE = \frac{1}{2n}\sum_{i=1}^{n}(x_i - g\left(f(x_i)\right)^2 \quad (3)$$

$$= \frac{1}{2n}\sum_{i=1}^{n}(x_i - \check{x}_i)^2$$

## III.8 Data classification:

After significantly reducing the number of attributes (features) of the original database, it is now possible to predict the classes of the new instances. For this, we decided to use a deep convolutional neural network CNN which is a class of deep neural networks, most commonly applied to analyzing visual imagery ,it uses a variation of multilayer

perceptrons designed to require minimal preprocessing ,the network consists of multiple layers of convolutions with nonlinear activation functions like ReLU or tanh that are applied to the results , a convolutional layer applies a convolution operation to the input, passing the result to the next layer [39].

### III.8.1 One-Dimensional Convolutional Neural Network (1D-CNN)

Is used primarily for the analysis of sequential data. The input to a 1D CNN is typically a time series of data or any data that has a sequence of data points,in a 1D CNN, convolutional layers operate over a 1D spatial domain, where the convolution filters slide along only one dimension to extract time-dependent features. This allows the network to capture temporal properties and patterns within the input data [39]

1D Convolutional Neural Networks can be used for tabular data, especially when the data includes sequential relationships within rows or columns, like time series datasets, each row or column can be treated as a sequence[40] .

Figure 16 shows a model that contains two convolutional layers, one flatten layer, two dense layers and one output layer , the function of One-Hot Encoding (One-of-K Coding) is to Transform categorical data into a binary vector representation.

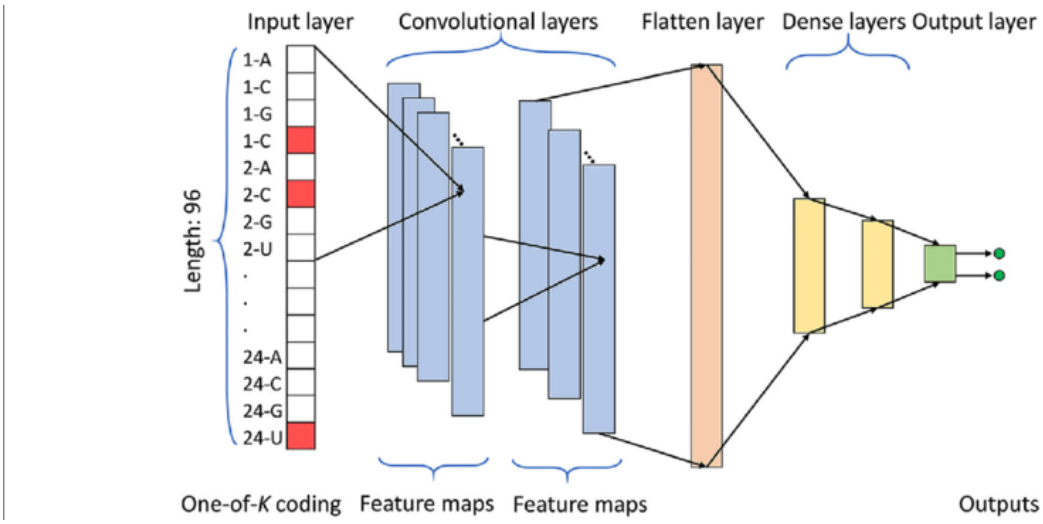



**Figure 16:Exemple of architecture of CNN 1D model[32].**

### III.8.2 Two-Dimensional Convolutional Neural Network (2D-CNN)

CNN 2D Model is primarily for matrix-like data like images , this type of CNN performs operations over 2D spatial data, processing inputs like images which are represented

as 2D matrices, the convolution layers in a 2D CNN use filters that are also 2D and slide over the image spatial dimensions (width and height) , this operation reduces the size of the image into a form that is easier to analyze, without sacrificing the relationships and patterns critical for making predictions[41] , CNN 2D Model can also be employed to tabular data by transforming the dataset into a 2D matrix format,this transformation allows the 2D convolutional layers to explore interactions across both rows and columns [42].

### III.8.2.1 Data Transformation for Applying 2D Convolutional Neural Networks

To apply a 2D Convolutional Neural Network (CNN) to tabular data, it is necessary to transform the data into a format compatible with the network's architecture. This process involves first determining the number of samples and features in the dataset. The features are then reshaped into a 2-dimensional grid. For a dataset with n_features, two factors of n_features are identified that are as close to each other as possible, approximating a square shape. This approach preserves the spatial relationships among the features when reshaped.

After determining the suitable 2D shape (height, width), the data is reshaped into a 4D tensor with the dimensions (n_samples, height, width, 1).Here, n_samples represents the number of data samples, and 1 indicates a single channel, similar to a grayscale image,this reshaping process enables the CNN to apply 2D convolutional filters to the tabular data, capturing local dependencies and feature interactions that may not be identified by traditional methods, Table 6 shows the Steps to reshape tabular data into 4D tensor to use it for CNN 2D classification:

**Table 6:**Steps to reshape tabular data into 4D tensor

| **Steps** | **Description** |
|---|---|
| Determine Dimensions | Determine the number of samples and features in the dataset. |
| Identify Factors | Find two factors of the number of features that approximate a square shape. |
| Reshape Data | Reshape the data into a 4D tensor with dimensions (n_samples, height, width, 1). |
| Apply 2D Convolution | Utilize 2D convolutional filters to capture local dependencies and feature interactions. |

## III.9 The Hyperparameters used for CNN classifications

- **Input layer**

Input layer is designed to process input data as one-dimensional sequences of features. These sequences are structured according to the model (sequence_length, 1), where the first dimension, representing the length of the sequence, indicates the total size of the entire data set, while the second dimension, equal to 1, emphasizes that each element of this sequence is associated with a single characteristic [43].

- **Activation Function**

Activation function plays a crucial role in neural networks by allowing each layer to process data in a non-linear manner. Without this feature, each layer would simply be a linear combination of inputs, thus limiting the complexity of the models that can be built. The introduction of a non-linear activation function, such as sigmoid, ReLU (Rectified Linear Unit), tanh, Leaky ReLU, MaxOut or ELU (Exponential Linear Unit), significantly increases the network capacity to learn and adapt to various types of data. These functions are chosen for their continuity and differentiability[44], thus facilitating the mathematical calculations necessary for the machine learning process.

- **Flatten Layer**

Flatten Layer comes after processing by the convolution and pooling layers. It transforms two-dimensional feature matrices into a one-dimensional vector[43], thus facilitating the connection between convolution layers and dense ones that require input in that specific form.

- **Dense Layers (Fully Connected Layers)**

In a neural network architecture, there are dense layers, also called fully connected layers, which follow the convolution layers. These layers are characterized by their ability to connect each neuron to all neurons of the previous layer, thus enabling sophisticated processing of the extracted features. They play a crucial role in high-level reasoning, using activation functions like ReLU to introduce non-linearity. The number of dense layers and the number of neurons in each can vary, providing significant flexibility for optimizing model performance. The last layer, also fully connected, is dedicated to data classification. It transforms the inputs into a

one-dimensional form to assign probabilities to different possible classes[43] , which constitutes the final output of the model.

- **Dropout Layers**

To avoid overfitting, we introduced dropout layers after each dense layer in our model. These layers work by randomly turning off a portion of the neurons during each update during training, which helps prevent the model from remembering the training data too precisely. Adjusting the dropout rate is a crucial aspect of configuring these layers. Additionally, this method allows deep neural networks to better handle the problem of overfitting[2], ensuring that they do not become too specialized in specific features of the training data, limiting their ability to generalize to new situations or to identify new variables in new data sets.

- **Output Layer**

Output layer of the model, which is at the end of the architecture, employs a sigmoid activation function, this function produces output in the form of probabilities, thus facilitating the performance of a binary classification task[43] , Table 7 summarizes hyperparameters of the proposed DL models:

**Table 7:**Hyperparameters of the proposed DL models*[35]*

| **Hyperparameter** | **Description** |
|---|---|
| Number of layers | Depth of the neural network. More layers can model more complex features, but can lead to overfitting. |
| Number of units per layer | Breadth of each layer. More units can capture more complex features, but increase the risk of overfitting. |
| Number of filters | Number of filters in a convolution layer. More filters allow the model to learn more complex patterns but increase the computational requirements |

| | |
|---|---|
| Filter size | Size of filters in a convolution layer. Small filters capture local patterns and large filters capture more global patterns. |
| Kernel size | Determines how many input data points each filter handles at a time. |
| Pooling size | Size of the pooling operation, affecting the amount of downsampling in the network. |
| Number of dense layers | Number of fully connected (dense) layers in a network. More layers can model complex patterns, but increase the risk of overfitting. |
| Number of neurons in dense layers | Number of units in fully connected (dense) layers. More neurons can capture complex patterns, but increase the computational requirements and risk of overfitting. |
| Batch size | Affects how often the weights in the network are updated. |
| Optimizer | Different optimization algorithms (such as SGD, Adam, and RMSprop) have different behaviors and may influence how quickly and effectively the model learns. |
| Number of epochs | Number of times the learning algorithm works through the entire training dataset. |
| Padding | Helps maintain the spatial dimensions of the input, allowing the network to learn from the edge and corner information. |

## III.10 Feed-Forward Deep Neural Network (FFDNN)

Feed-Forward Deep Neural Network (FFDNN) is a type of artificial neural network, this model is characterized by its acyclic nature, the information flows in one direction from the input layer, passing by hidden layers to reach the output layer, the connections between the nodes do not loop back.

Each layer is fully connected to the next layer in order to approximate complex functions by learning hierarchical representations of the input data

various activation functions are used in this network such as ReLU and sigmoid to introduce the non-linearity, and Regularization techniques like dropout and batch normalization to prevent overfitting and enhance performance, Figure 17 shows Multi layer feed-forward network:

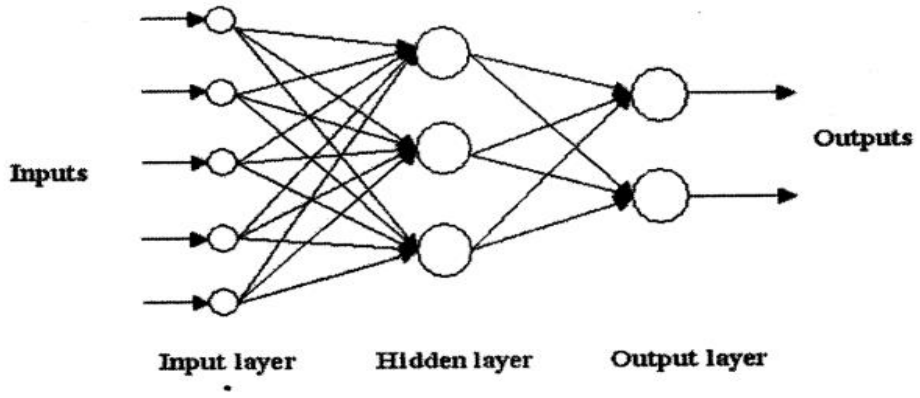



**Figure 17:Multi layer feed-forward network[37]**

Unidirectional flow is shown which is the characteristic of the feedforward networks, fully connected architecture is illustrated , every node in the input layer is connected with every node in the output layer , neurons and weights are used to pass from the input signals to the output layer to finally calculate the output signals.

According to S.Haykin [13]using more hidden layers enable the network to extract higher-order statistics, the function of hidden neurons is to intervene between the external input and the network output in some useful manner.

## III.11 Long Short-Term Memory (LSTM)

The central concept of LSTMs is their ability to translate and store inputs using memory cells over time, which replace the hidden units of recurrent networks and have recurrent connections. This memory cell will be manipulated by gates whose the activation function is defined by specific functions. Figure 18 clearly illustrates that Long Short-Term Memory (LSTM) networks consist of four types of gates: the forget gate, the update gate, the tanh gate, and the output gate, within these architectures learning occurs through adjustment of weights and activation function values, to enable efficient production of temporal features between inputs and outputs.

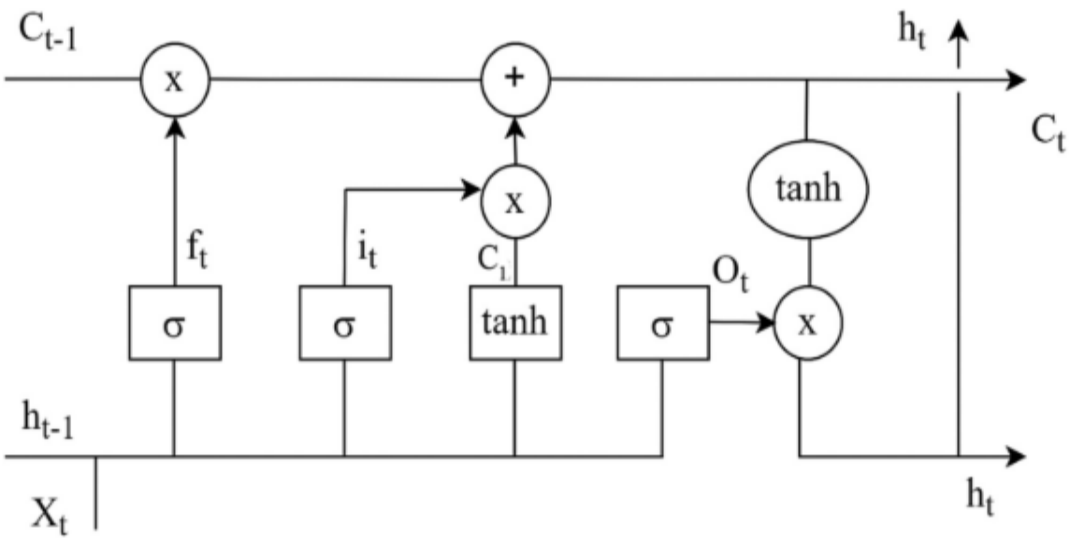



**Figure 18: General LSTM architecture[38].**

In LSTM network, the input and output values are the vectors of the same size and the calculations regarding the cell state, forget gates, input and output of the LSTM are given in the equations below -below. The forgetting gate $f_{(t)i}$ for cell i at time t will decide which information to keep and which to discard not via a sigmoid activation function (r) in the equation. 1. $b_f$ , $Z_f$ and $D_f$ indicate the gap, input weight and recurrent weights for the forget gates, respectively. Additionally, the output will be generated based on the sigmoid function and multiplied by the previous state of the cell. The cell state update gate $n_{(t)i}$takes into account the input gate, which will determine the necessary information to add to generate the output as demonstrated in equation 2 while $b$ , $Z$ and $D$ indicate the deviation, input weight and recurring weights entering the LSTM cell, respectively. In the equation 3, a calculation of the input gate for cell $p_{(t)i}$ is demonstrated and this calculation is performed in the same way as the forget gate. This generation will be based on the sigmoid function and the tanh function according to the tanh gate. The multiplication of these gates will be added to the output resulting from the multiplication of the forgotten gate with the previous state of the cell to generate the output. The current state of the cell passes through the tanh activation function, then is multiplied by the output of the sigmoid activation function of the output gate to generate the current hidden state, representing the output of the LSTM network. In the equation 4, $h_{(t)i}$ indicates the hidden state while , $s_{(t)i}$ indicates the output gate. The output gate equation is demonstrated in Equation 5 while $b'$, $Z'$and $D'$ indicate the gap, input weight and recurrent weights respectively[45].

$$f_i^{(t)} = \sigma(b_i^f + \sum jZ_{i.j}^f x_j^{(t)} + \sum jD_{i.j}^f h_j^{(t-1)}) \quad (4)$$

$$n_i^{(t)} = f_i^{(t)} n_i^{(t-1)} + p_i^{(t)} \sigma\left(b_i + \sum jZ_{i.j}\, x_j^{(t)} + \sum jD_{i.j}\, h_j^{(t-1)}\right) \quad (5)$$

$$p_i^{(t)} = \sigma(b_i^p + \sum jZ_{i.j}^p x_j^{(t)} + \sum jD_{i.j}^p h_j^{(t-1)}) \quad (6)$$

$$h_i^t = \tanh\left(n_i^{(t)}\right) s_i^{(t)} \quad (7)$$

$$s_i^{(t)} = \sigma(b_i^0 + \sum jZ_{i.j}^0 x_j^{(t)} + \sum jD_{i.j}^0 h_j^{t-1} \quad (8)$$

## III.12 Visual Geometry Group (VGG)

VGG16, developed by Simonyan and Zisserman from the University of Oxford's Visual Geometry Group (VGG), is a convolutional neural network (CNN) model that participated in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2014. The architecture of VGG16 consists of 16 layers, including 13 convolutional layers and 3 fully connected layers. It employs very small receptive fields of 3x3 filters, effectively capturing essential spatial hierarchies in images. Max-pooling is performed over a 2x2 pixel window with a stride of 2. The depth of VGG16 enables the model to learn complex features, achieving a reduced error rate of 7.3% on the ImageNet challenge. Due to its simplicity and effectiveness, VGG16 is widely utilized in image classification, object detection, and image segmentation tasks[46], Figure 19 shows VGG16 Architecture.

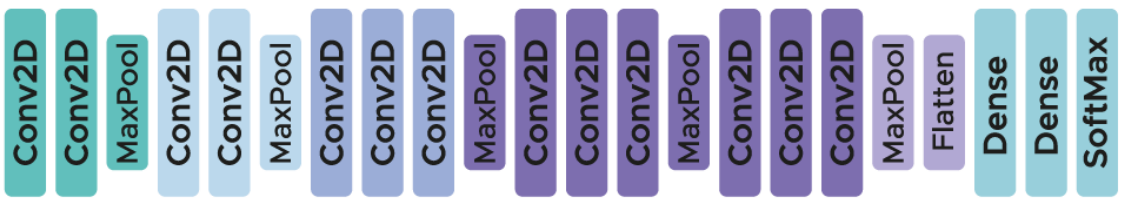



**Figure 19:Vgg16 Architecture[45]**

## III.13 MobileNet V2

[5] MobileNet V2, designed for mobile and embedded vision applications, represents an advancement over the original MobileNet model. It features an inverted residual structure where shortcut connections are made between thin bottleneck layers and employs lightweight depthwise convolutions in the intermediate expansion layer, enhancing computational efficiency. Key innovations of MobileNet V2 include linear bottlenecks and inverted residual blocks, which help maintain high-dimensional feature spaces while reducing parameters and computational complexity. Despite its compact size, MobileNet V2 demonstrates high accuracy on large-scale image classification tasks, including benchmarks like ImageNet[47].

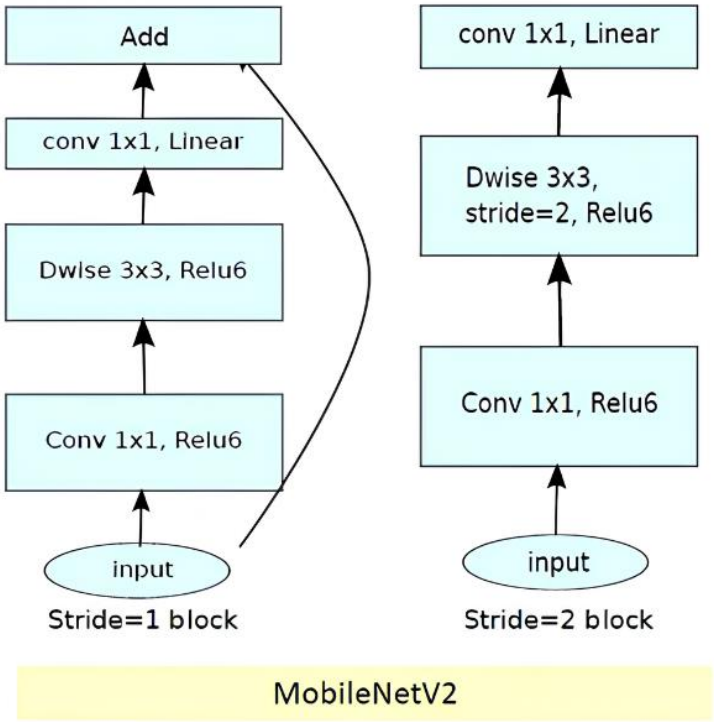



**Figure 20:Architecture of MobilenetV2**

[5] https://www.researchgate.net/figure/Architecture-of-MobilenetV2_fig6_374547650

## III.14 DenseNet 121

DenseNet 121, or Densely Connected Convolutional Networks, is a type of convolutional neural network (CNN) that introduces direct connections between any two layers with the same feature-map size. Unlike traditional CNNs, which connect each layer only to the next one, DenseNet121 connects each layer to every other layer in a feed-forward fashion, enhancing feature propagation, improving gradient flow, and reducing the number of parameters while maintaining high performance. This architecture consists of 121 layers, making it a deep network suitable for complex image classification tasks[48].

## III.15 Performance Indicators

Regarding model evaluation, many metrics have been developed to estimate the accuracy and robustness of different machine learning and deep learning algorithms ,these metrics allow us to better understand the performance of the models, to evaluate the performance of our network intrusion detection model based on deep learning architectures, we used a combination of metrics including precision, recall, F1 score and accuracy, they are calculated from the values obtained from the confusion matrix, presented in Figure 21 ,in this matrix, TP represents true positives, FN false negatives, FP false positives and TN true negatives .

- **Confusion matrix :** Is widely used table to visualize classification performance method ,its rows represent the actual classes, while its columns represent the predicted classes.

| | Predicted Attack | Predicted Normal |
|---|---|---|
| Actual Attack | TP | FN |
| Actual Normal | FP | TN |

**Figure 21:Confusion matrix [2]**

- **Accuracy:** It measures a system's ability to correctly identify both normal and harmful traffic [49]. It is calculated using the following formula:

$$\text{Accuracy} = \frac{TP+TN}{TP+FP+FN+TN} \qquad (9)$$

- **Recall :**Detection rate (DR) known Recall and also named Sensitivity and true positive rate, it is number of elements correctly identified as positive compared to the total true positives , it measures the proportion of actual intrusions that were correctly identified[49] ,this measurement is given by the next formula:

$$\text{Recall} = \frac{TP}{TP+FN} \qquad (10)$$

- **Precision :** It measures the proportion of correctly identified intrusions out of all predicted intrusions [49]. It is calculated as following formula :

$$\text{Precision} = \frac{TP}{TP+FP} \qquad (11)$$

- **F1 score:** It provides a balance between precision and recall, especially useful when the class distribution is imbalanced. it is computed as the harmonic mean of the model's precision and recall, and provides a single value that combines both measures ,by integrating precision and recall, the F1 score can provide a more comprehensive evaluation of a model's effectiveness in accurately identifying true positives while minimizing false positives and false negatives[38] , this measurement is given by the next formula:

$$\text{F1} = \frac{2\times(\text{Recall}\times\text{Precision})}{\text{Recall}+\text{Precision}} \qquad (12)$$

A high F1-score indicates a model with low false positives and false negatives.

- **Train and validation accuracy plot** : is a graphical representation used in machine learning to evaluate a model's performance during the training process, this plot typically shows the accuracy of the model on both the training dataset and the validation dataset over a number of training epochs.

  Figure 22 shows Training and Validation Accuracy, Blue line on the graph shows how accurately the model predicts the training data. It helps to understand if the model is learning from the training examples, the Orange line shows how accurately the model predicts data from a validation set, which is not seen by the model during training. It helps to understand how well the model generalizes :

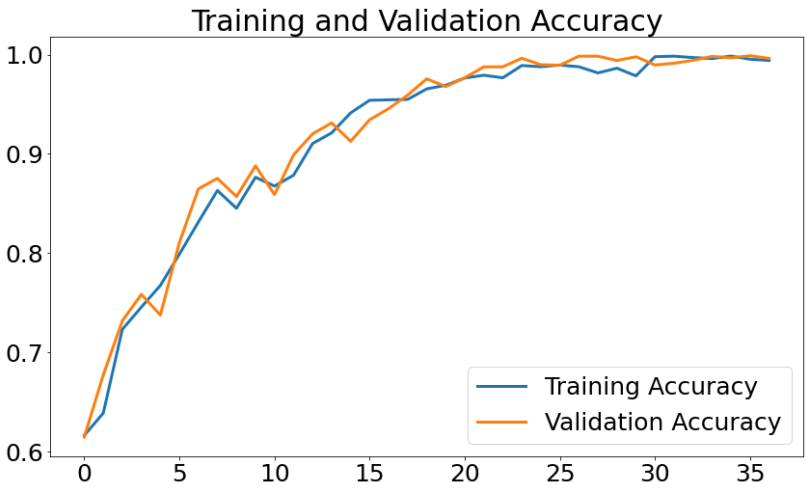



**Figure 22:Example of Training and validation accuracy Graph[42]**

# IV Chapter IV

# Experimental Results and Discussion

# IV.1 Introduction

This chapter summarizes and analyzes the evaluation results in terms of attack detection and attack data generator performance. In more detail, we first introduce by the Conception of The proposed Deep Learning-based Intrusion Detection systems for IoT networks in python and evaluation metrics in Section A. The detailed evaluation results and discussions are presented in Section B.

In our study, we developed the the models and trained them using a combination of local and cloud-based resources to leverage computational efficiency, the project was trained on a local machine equipped with an Intel i5 6th generation processor, 12GB of RAM, and a 256GB SSD, this setup provided a robust environment for preliminary model development and testing ,however, to handle larger datasets and more complex computations it needs high-memory RAM capabilities , and to execute VGG16 MobileNetV2 and Densent-121 model we need GPU for that we used Google Colab Pro the use of TensorFlow for model training, allowing for enhanced performance and scalability.

# IV.2 Experimental Datasets, Deep Learning Models used and Evaluation Metrics

In the experiments, we used the CICIDS2017 and CICDDOS2019 datasets to evaluate detection and runtime performance, due to their popularity and contains a sufficient amount of anomalous network traffic from common cyber attacks to result in deep learning models. Moreover because they are the latest in cybersecurity, aiming to minimize the drawbacks of popular datasets (e.g., KDDCUP, KDD99, NSL-KDD),we trained our intrusion detection system (IDS) by various deep learning models, including Autoencoder, 1D-CNN, 2D-CNN, FFDNN ,LSTM and VGG16 combined with MobileNetV2 and DenseNet 121.

To evaluate the effectiveness of the attack detection models, improved via hyperparameter tuning, we used several metric evaluations, including confusion matrix, accuracy, precision, recall, and F1 score. Let TP, TN, FP and FN denote true positives, true negatives, false positives and false negatives. Together, these indicators provide an overview of the model's effectiveness in accurately detecting network intrusions while minimizing the false positive and false negative errors.

## IV.3 Data Preprocessing

The initial step involved data cleaning, where we removed NaN values and duplicate rows. For binary classification, we introduced a new column, "Class," derived from the "Label" column. If the "Label" was "Benign," the "Class" was set to "Benign"; otherwise, it was set to "Attack." Both the "Label" and "Class" columns were then encoded to numerical values for further processing.

Subsequently, all features were scaled to a range between 0 and 1 to facilitate the application of deep learning models, we used all the rows of CicIDS2017 dataset after cleaned it we obtained 2 520 798 rows , and we used a part of CicDDOS2019 dataset after cleaned it we obtained 1 406 277.

Next step was assess the balance of the dataset, and we found it to be imbalanced, initial attempts to work with the imbalanced dataset gived unsatisfactory results, the accuracy was augmented but the precision and recall gave a bad results,we tried to use a 1D CNN model for classification without balancing the data and obtained the following results: Accuracy: 96%, Recall: 50%, Precision: 52%.

These results indicate that the model is favoring one class over the other, leading to many false positives and false negatives,this imbalance suggests the need for data balancing techniques to improve the model's performance across all classes.
Therefore, we employed the Synthetic Minority Over-sampling Technique (SMOTE) to balance the dataset, next table shows the number of the rows of cicIDS2017 dataset before and after balancing:

**Table 8:**Number of rows before and after balancing the dataset

| | CicIDS2017 Dataset | | CicDDOS2019 Dataset | |
|---|---|---|---|---|
| | Number of rows | Number of each Class | Number of rows | Number of each Class |
| Before balancing | 2 520 798 | Class“0”:2 095 057<br>Class”1”: 425 741 | 1 756 323 | Class“0”: 1,706,323<br>Class”1”: 50 000 |

| | | | | |
|---|---|---|---|---|
| After Balancing | 4 190 114 | Class“0”:2 095 057<br>Class”1”:2 095 057 | 2 712 554 | Class“0”:1 356 277<br>Class”1”:1 356 277 |

The final preprocessing step involved splitting the dataset into training and testing sets, allocating 80% of the data for training and 20% for testing. With these preprocessing steps completed, the dataset was ready for classification.

## IV.4 Feature Extraction

Reducing number of features minimizes the complexity of the model, leading to faster training and inference times,this is especially important for handling large datasets and real-time intrusion detection, when using fewer features models can focus on the most relevant information, potentially improving detection accuracy and reducing the risk of overfitting , this step eliminate redundant and irrelevant information, reducing noise in the data. Extract features is crucial for deploying models in resource-limited environments because lower-dimensional data is easier to manage and requires less computational power, this efficiency gain ,

### IV.4.1 Extracting Features Using Autoencoder

In our approach, we used an autoencoder for feature extraction,the autoencoder plays a crucial role in reducing the number of features in the dataset while retaining the most important information,this step is essential for improving the efficiency and performance of our intrusion detection models, for the CICIDS2017 Dataset the autoencoder reduced the number of its features from 78 to 64, and for CICDDoS2019 Dataset the autoencoder reduced the number of its features from 87 to 64

#### IV.4.1.1 Autoencoder parameters

Next table describes the format of the parameters and output shapes of our autoencoder model:

**Table 9:** Parameters and output shapes of the autoencoder model

| **Step** | **Parameters Description** |
|---|---|
| Imports and Data Preparation | Import libraries (pandas, numpy, tensorflow, sklearn, matplotlib).<br>Load the dataset (X_train, X_test,y_train,y_test). |

| | |
|---|---|
| Define Input Layer | Input layer with shape matching the number of features in X_train. |
| Encoding Layers | 1. Dense layer with 128 neurons, ReLU activation, followed by batch normalization and dropout (0.5).<br>2. Dense layer with 64 neurons, ReLU activation, followed by batch normalization and dropout (0.5).<br>3. Dense layer with 64 neurons, ReLU activation (encoding dimension). |
| Decoding Layers | 1. Dense layer with 64 neurons, ReLU activation, followed by batch normalization and dropout (0.5).<br>2. Dense layer with 128 neurons, ReLU activation, followed by batch normalization and dropout (0.5).<br>3. Dense layer with the same number of neurons as input, sigmoid activation. |
| Compile the Model | Optimizer: Adam<br>Loss function: mean squared error (mse). |
| Early Stopping | Used early stopping callback to monitor validation loss with patience of 5 epochs, restoring best weights. |
| Train the Autoencoder | Train on X_train, validate on X_test with early stopping.<br>Batch<br>size: 256<br>Epochs: 20 |
| Plot Training & Validation Loss | Plot the training and validation loss over epochs using matplotlib. |

The autoencoder model was used to extract 64-dimensional features from the training and testing datasets. Initially, the encoder part of the autoencoder was employed to predict the encoded representations of the input data. Specifically, the X_train and X_test datasets were passed through the encoder, resulting in X_train_encoded and X_test_encoded, respectively. These encoded features, each comprising 64 dimensions, were used for the input of the next classification models.

### IV.4.1.2 Autoencoder Evaluation

To evaluate the performance of the Autoencoder, we employed the Mean Squared Error (MSE) as our primary metric. MSE measures the average of the squares of the errors, indicating the difference between the observed actual outcomes and the outcomes predicted by the model.

The training and validation performance of the Autoencoder is illustrated in Figure 23. As shown in the figure, the autoencoder model demonstrated robust performance, with a Mean Squared Error (MSE) of 0.0023on the training set and 0.00126 on the validation set over the course of 18 epochs. The close alignment of MSE values between the training and validation sets indicates strong generalization to unseen data and minimal risk of overfitting or underfitting, next Figure shows MSE for both training and validation sets over 17epochs:

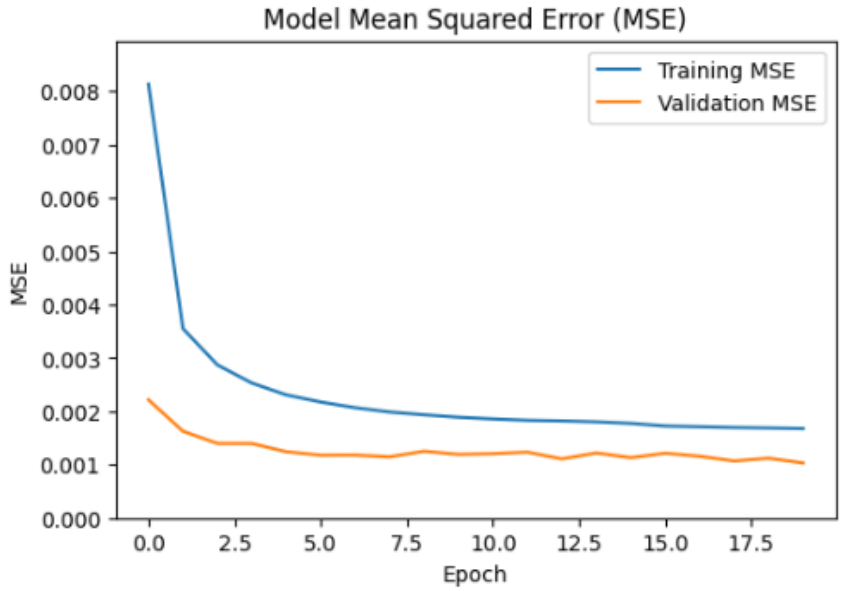



**Figure 23:MSE for both training and validation sets over 17 epochs**

The low MSE values underscore the model's proficiency in capturing and reconstructing the underlying data patterns with minimal information loss. This capability is critical for efficient feature extraction, ensuring that the most significant features are preserved for subsequent intrusion detection tasks.

The model's effectiveness in reducing data dimensionality while preserving high fidelity enhances both the efficiency and speed of training and inference in downstream intrusion detection models,this balance between model complexity and generalization is pivotal for real-time intrusion detection systems, ensuring reliable performance on new, unseen data.

In summary, the autoencoder's low MSE values and minimal discrepancy between training and validation sets confirm its efficacy in feature extraction and dimensionality reduction, thereby bolstering the performance and operational efficiency of intrusion detection models in practical deployments.

### IV.4.2 Extracting Features Using VGG16

Feature extraction using VGG16 involves leveraging its robust architecture to capture intricate patterns and features from input data. VGG16, a renowned deep convolutional neural network developed by the Visual Geometry Group at the University of Oxford, is celebrated for its effectiveness in various computer vision tasks. The network consists of 16 layers, including convolutional and fully connected layers, which excel in learning hierarchical features from images. By utilizing only the convolutional layers of VGG16 and omitting the top fully connected layers, we can extract high-level features from the input data. These features are then processed using a global average pooling layer, which reduces the spatial dimensions, making them suitable for subsequent classification tasks. This method ensures that the extracted features are both rich and representative, encapsulating essential information necessary for accurate classification[46].

We extracted the feature using VGG16, excluding the top layers, and appended a global average pooling layer to condense the spatial features into a lower-dimensional feature vector ,this approach ensured that the extracted features captured critical information while maintaining computational efficiency.

The VGG16 features extractor was configured with the top layers excluded (`Include Top: False`) and without pre-trained weights (`Weights: None`). The output layer of the extractor is a GlobalAveragePooling2D layer.

The features extracted by Vgg16 were used in Mobilnet V2 and DenseNet-121 combination classification model .

## IV.5 Classification using Deep Learning Models

### IV.5.1 AE+CNN 1D Model

Convolutional Neural Networks (CNNs) are highly proficient in the automatic extraction of pertinent features from raw input data. This capability significantly diminishes

the necessity for manual feature engineering, thereby enabling the model to discern intricate patterns and correlations within network traffic data that may indicate potential intrusions[33]

So used CNN 1D model for binary classification, next table describes the parameters and output shapes of the CNN 1D model:

**Table 10:**Parameters and output shapes of the CNN 1D model

| Layer Type | Output Shape | Parameters |
|---|---|---|
| Conv1D Layer (conv1d_2) | (None, 62, 32) | 128 |
| MaxPooling1D (max_pooling1d_2) | (None, 31, 32) | 0 |
| Conv1D Layer (conv1d_3) | (None, 29, 64) | 6208 |
| MaxPooling1D (max_pooling1d_3) | (None, 14, 64) | 0 |
| Flatten Layer (flatten_1) | (None, 896) | 0 |
| Dense Layer (dense_2) | (None, 78) | 69966 |
| Dropout Layer (dropout_1) | (None, 78) | 0 |
| Dense Layer (dense_3) | (None, 1) | 79 |

Next table shows the results of the model on two datasets:

**Table 11:**Results of AE+CNN 1D Model on the CicIDS 2017 dataset and CicDDos 2019 dataset

| **Metric** | **CicIDS 2017 Dataset** | **CicDDoS 2019 Dataset** |
|---|---|---|
| **Accuracy** | 97.99% | 97.69% |
| **Recall** | 96.57% | 97.93% |
| **Precision** | 99.39% | 97.54% |
| **F1-Score** | 97.96% | 97.74% |

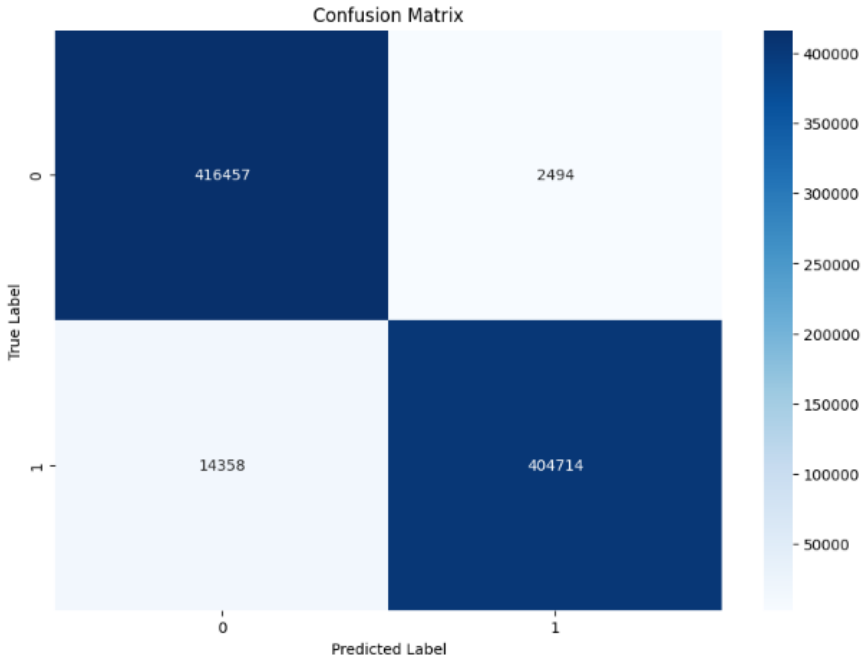



**Figure 24:Confusion Matrix of AE+CNN 1D Model on CicIDS2017 Dataset**

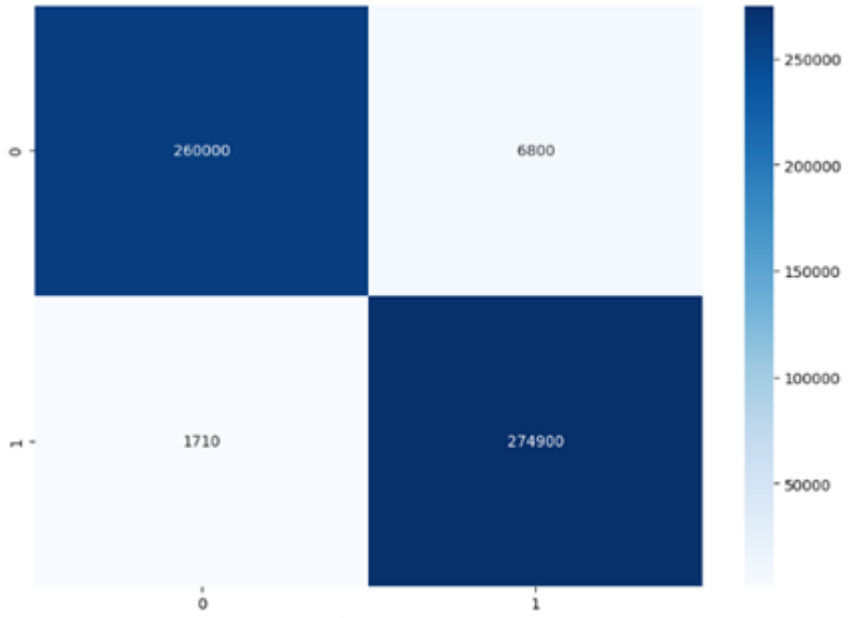



**Figure 25:Confusion Matrix of AE+CNN 1D Model on CicDDOS2019 Dataset**

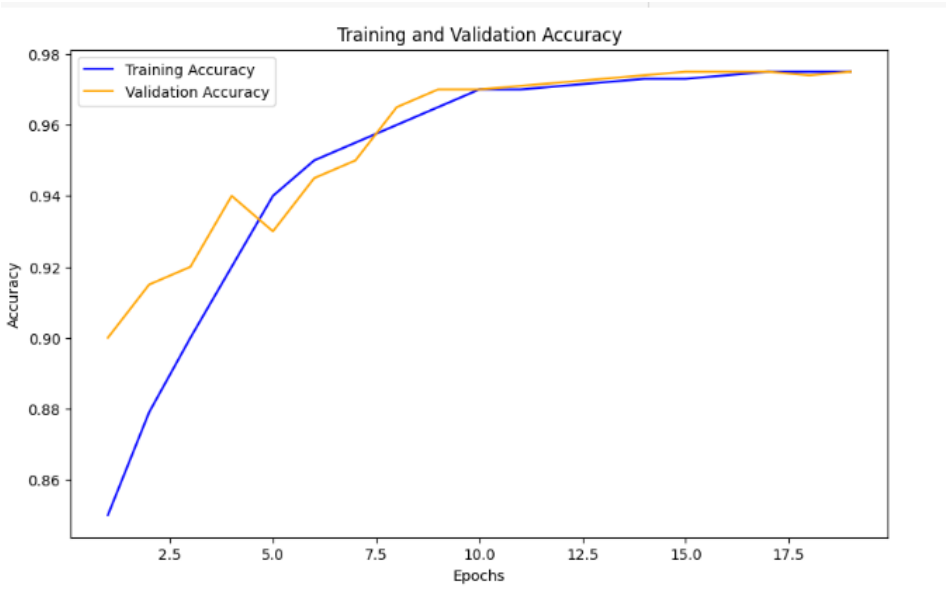



**Figure 26:Training and Testing Accuracy Graph of AE+CNN 1D Model on CicIDS2017 Dataset**

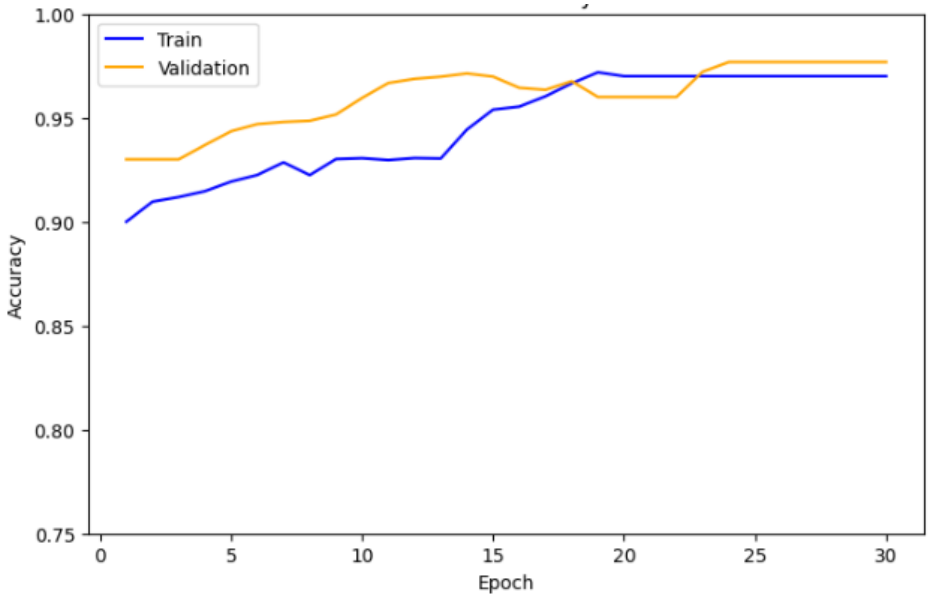



**Figure 27:Training and Testing Accuracy Graph of AE+CNN 1D Model on CicDDOS2019 Dataset**

### IV.5.2 AE+CNN 2D Model

CNN 2D Model used the features extracted by the autoencoder for binary classification on CicIDS2017 dataset and CicDDOS 2019 datase next table describes the format of the parameters and output shapes of the model created:

**Table 12:**Parameters and output shapes of the CNN 2D model

| Layer Type | Output Shape | Parameters |
|---|---|---|
| Conv2D Layer (conv2d) | (None, 8, 8, 32) | 320 |
| MaxPooling2D (max_pooling2d) | (None, 4, 4, 32) | 0 |
| Conv2D Layer (conv2d_1) | (None, 4, 4, 64) | 18496 |
| MaxPooling2D (max_pooling2d_1) | (None, 2, 2, 64) | 0 |
| Flatten Layer (flatten) | (None, 256) | 0 |
| Dense Layer (dense) | (None, 128) | 32896 |
| Dropout Layer (dropout) | (None, 128) | 0 |
| Dense Layer (dense_1) | (None, 1) | 129 |

Next table shows the results of the model on two datasets:

**Table 13**:Results of AE+CNN 2D Model on the CicIDS 2017 dataset and CicDDos 2019 dataset

| **Metric** | **CicIDS 2017 Dataset** | **CicDDoS 2019 Dataset** |
|---|---|---|
| **Accuracy** | 98,23% | 98.43% |
| **Recall** | 96,82% | 99.38% |
| **Precision** | 99,63% | 97.59% |
| **F1-Score** | 98,20% | 98.48% |

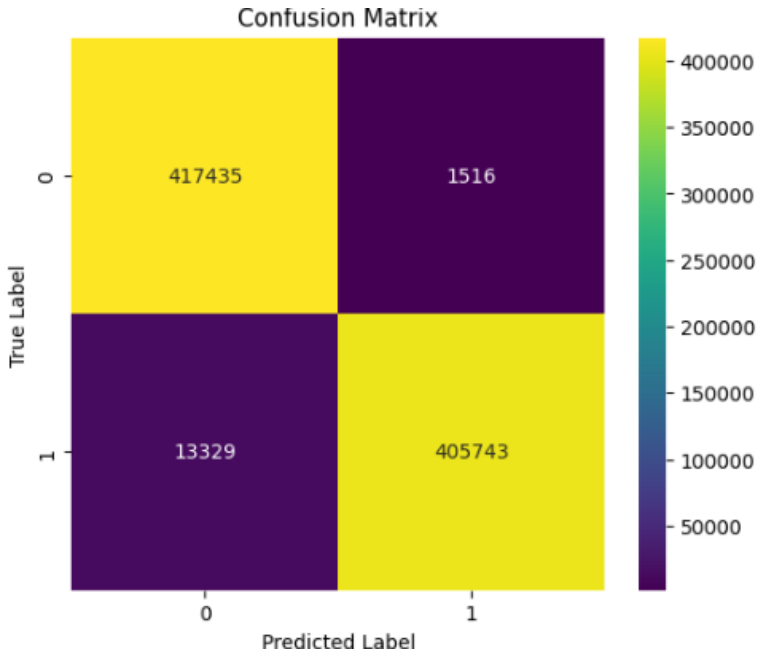



**Figure 28:Confusion Matrix of AE+CNN 2D Model on CicIDS2017 Dataset**

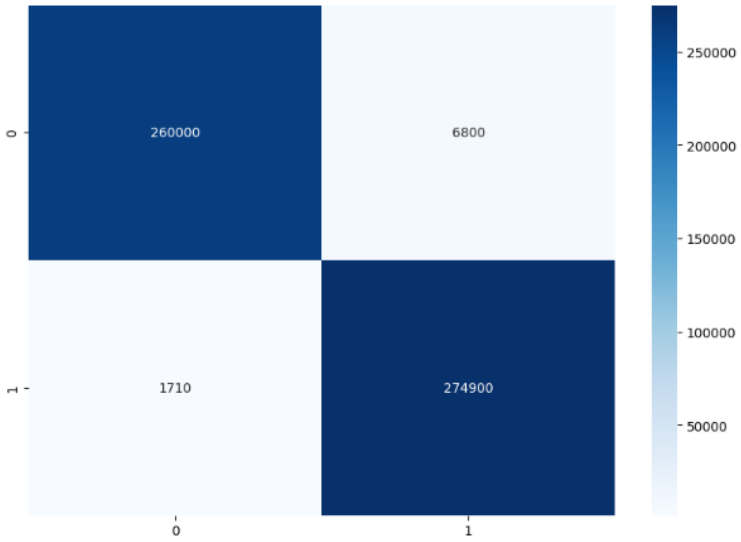



**Figure 29:Confusion Matrix of AE+CNN 2D Model on CicDDOS2019 Dataset**

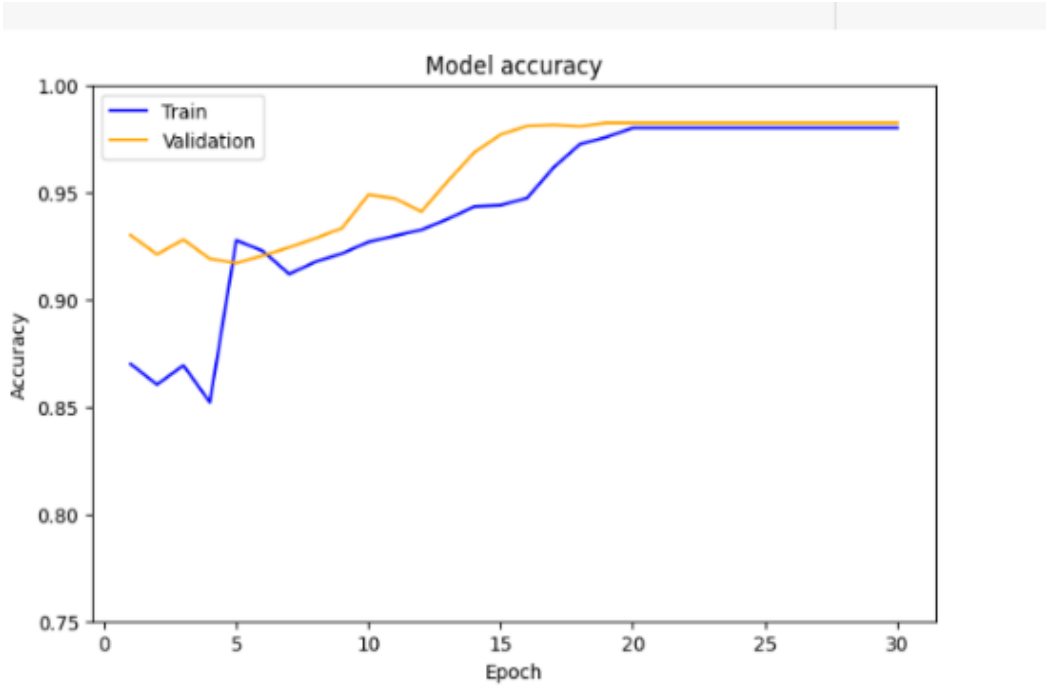



**Figure 30:Train and Testing Accuracy Graph of AE+CNN 2D Model**

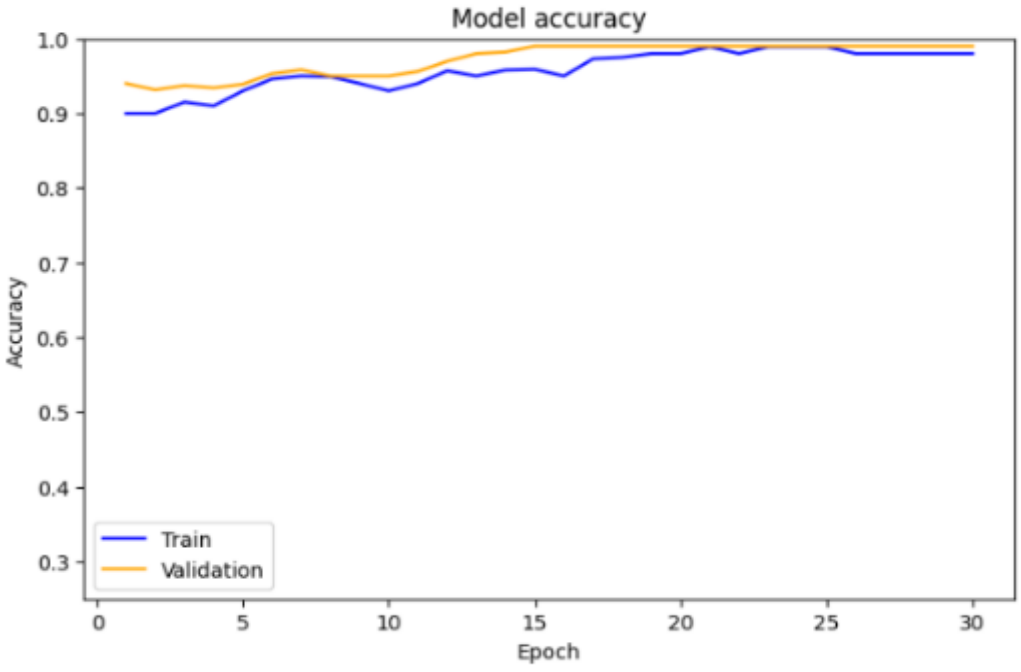



**Figure 31:Train and Testing Accuracy Graph of AE+CNN 2D Model**

### IV.5.3 AE+FFDNN Model

FFDNN Model used the features extracted by the autoencoder for binary classification on CicIDS 2017 dataset, next table describes the format of the parameters and output shapes of the model created:

**Table 14:**Parameters and Output shapes of the FFDNN Model

| Layer Type | Output Shape | Parameters |
|---|---|---|
| Dense Layer (dense) | (None, 128) | 8320 |
| Batch Normalization (batch_normalization) | (None, 128) | 512 |
| Activation (activation) | (None, 128) | 0 |
| Dropout (dropout) | (None, 128) | 0 |
| Dense Layer (dense_1) | (None, 256) | 33024 |
| Batch Normalization (batch_normalization_1) | (None, 256) | 1024 |
| Activation (activation_1) | (None, 256) | 0 |
| Dropout (dropout_1) | (None, 256) | 0 |
| Dense Layer (dense_2) | (None, 512) | 131584 |
| Batch Normalization (batch_normalization_2) | (None, 512) | 2048 |
| Activation (activation_2) | (None, 512) | 0 |
| Dropout (dropout_2) | (None, 512) | 0 |
| Dense Layer (dense_3) | (None, 256) | 131328 |
| Batch Normalization (batch_normalization_3) | (None, 256) | 1024 |
| Activation (activation_3) | (None, 256) | 0 |
| Dropout (dropout_3) | (None, 256) | 0 |
| Dense Layer (dense_4) | (None, 128) | 32896 |
| Batch Normalization (batch_normalization_4) | (None, 128) | 512 |

| | | |
|---|---|---|
| Activation (activation_4) | (None, 128) | 0 |
| Dropout (dropout_4) | (None, 128) | 0 |
| Dense Layer (dense_5) | (None, 1) | 129 |

Next table Shows the results of this model on CicIDS2017 dataset :

**Table 15:**Results of FFDNN Model on CicIDS2017

| Metric | CicIDS 2017 Dataset |
|---|---|
| **Accuracy** | 97,68% |
| **Recall** | 95,83% |
| **Precision** | 99,50% |
| **F1-Score** | 97,63% |

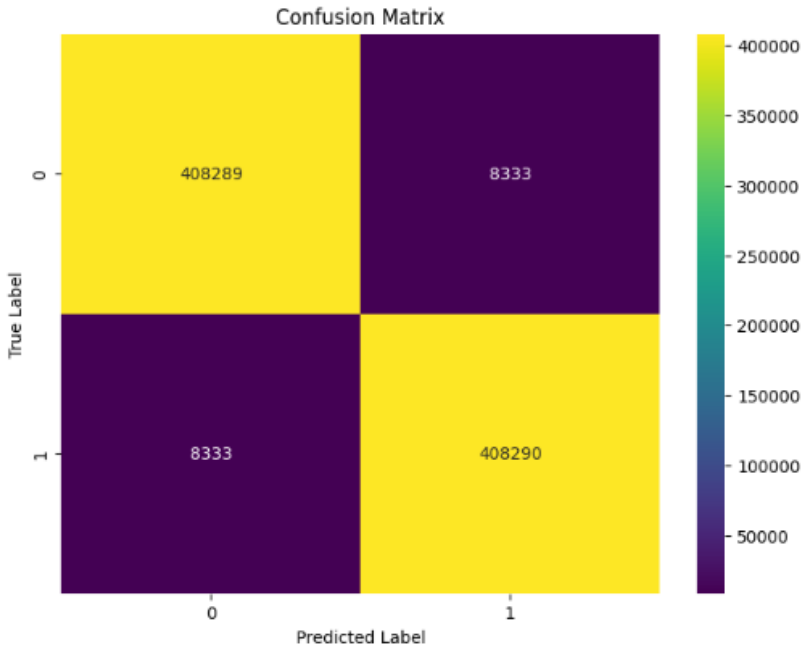



**Figure 32:Confusion matrix of FFDNN Model on CicIDS2017 dataset**

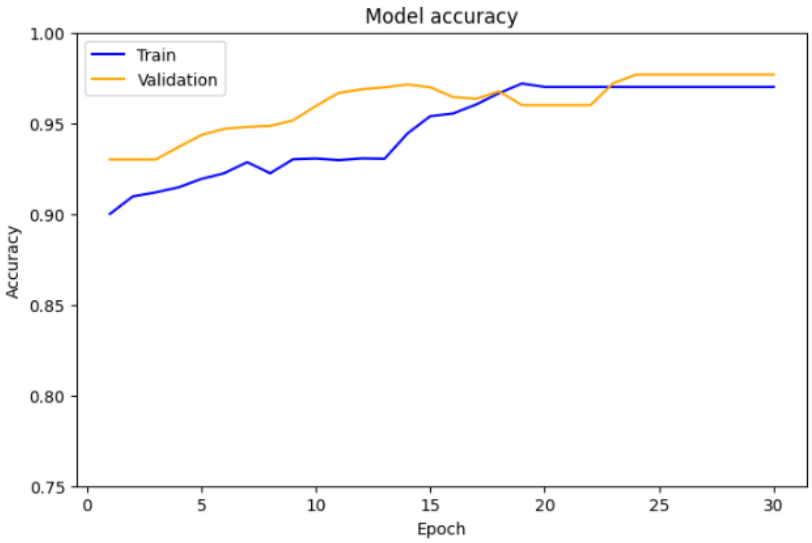



**Figure 33:Train and Testing Accuracy Graph of AE+FFDNN Model**

## IV.5.4 AE+LSTM Model

Features extracted by the autoencoder are used by LSTM model for binary classification on CicIDS 2017 dataset, next table describes the format of the parameters of FFDNN model created:

**Table 16**:Parameters of LSTM model created

| Parametrs | Values |
|---|---|
| **Model Type** | Sequential |
| **Layer 1** | LSTM |
| **Units (Layer 1)** | 100 |
| **Input Shape (Layer 1)** | (64, 1) |
| **Layer 2** | Dense |
| **Units (Layer 2)** | 1 |
| **Activation (Layer 2)** | Sigmoid |

| | |
|---|---|
| **Optimizer** | Adam |
| **Loss Function** | Binary Cross-Entropy |
| **Batch Size** | 256 |
| **Epochs** | 30 |

Following Table shows Results of AE+LSTM Model on the CicIDS 2017 dataset:

**Table 17:** Results of AE+LSTM Model on the CicIDS 2017 dataset

| **Metric** | **CicIDS 2017 Dataset** |
|---|---|
| **Accuracy** | 98.01% |
| **Recall** | 98,09% |
| **Precision** | 98.32% |
| **F1-Score** | 98,20% |

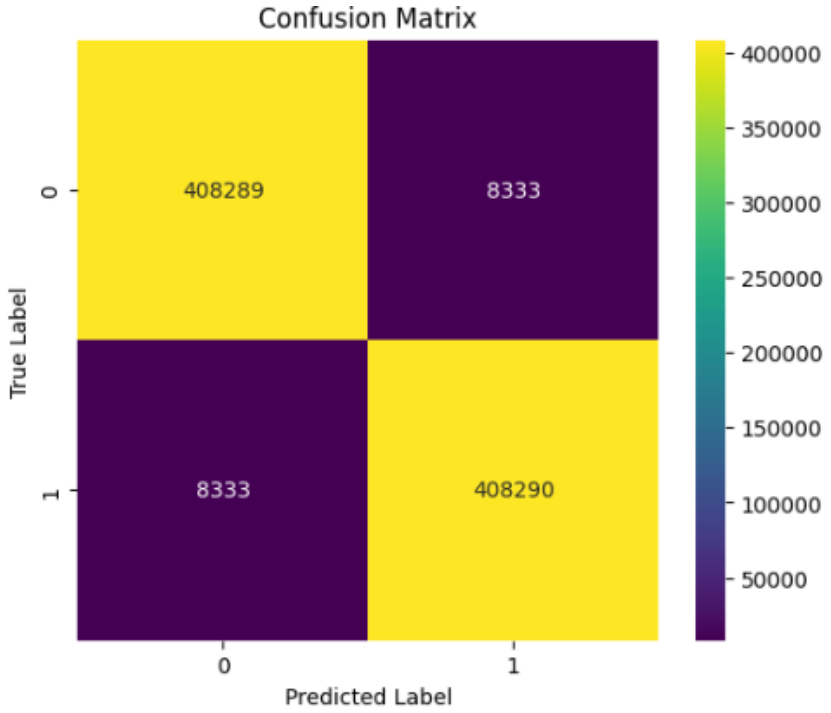



**Figure 34:Confusion Matrix of AE+LSTM Model on CicIDS2017 Dataset**

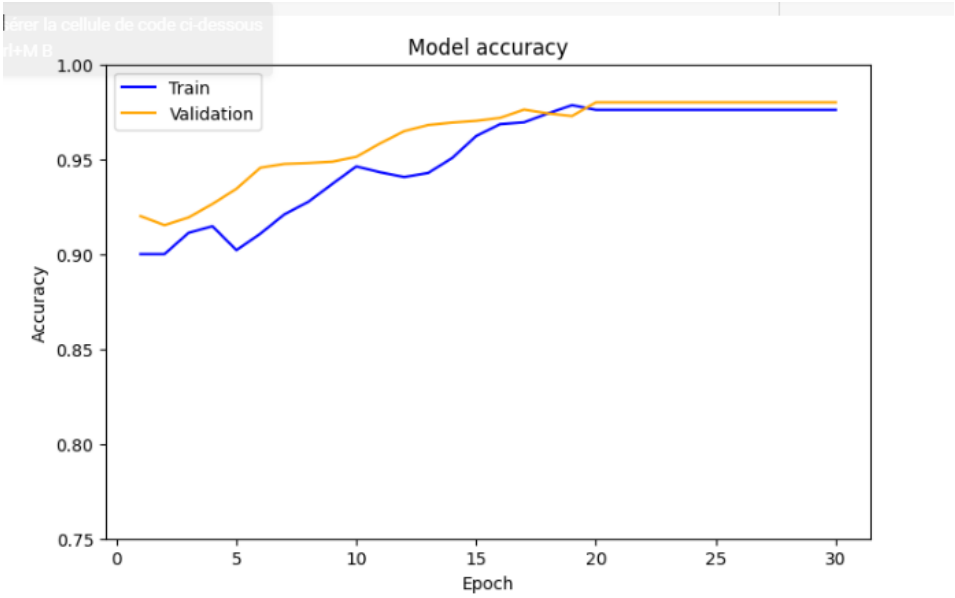



**Figure 35:Train and Testing Accuracy Graph of AE+LSTM Model**

### IV.5.5 Combination of Vgg16 , MobileNetV2 and DenseNet121 Model

we built classification models using MobileNetV2 and DenseNet121 for binary classification tasks ,the architecture of each classification model included a fully connected dense layer with 256 units and ReLU activation, followed by a sigmoid output layer to provide binary classification probabilities. The models were compiled using the Adam optimizer with a learning rate of 0.001, and the loss function used was binary cross-entropy.

Early stopping was implemented during training to prevent overfitting, monitoring the validation loss with a patience parameter of 3 epochs. This ensured that the best weights were restored for the final evaluation, next table define the parameters and values used for models used:

**Table 18:** Parameters and values of the model component

| **Model Component** | **Parameters** | **Values** |
|---|---|---|
| Data Preprocessing | Input Shape | 80 features |
| | Padding | Zero padding to 80 |
| | Reshape | 8x10 |
| | Resize | 32x32x1 |
| Feature Extractor (VGG16) | Include Top | False |

| | | |
|---|---|---|
| | Weights | None |
| | Output Layer | GlobalAveragePooling2D |
| Classification Layer | Dense Layer Units | 256 |
| | Dense Layer Activation | ReLU |
| | Output Layer Units | 1 |
| | Output Layer Activation | Sigmoid |
| Model Compilation | Optimizer | Adam |
| | Learning Rate | 0.001 |
| | Loss Function | Binary Cross-Entropy |
| Training | Early Stopping Monitor | Validation Loss |
| | Early Stopping Patience | 3 |
| | Epochs | 20 |

After extracting features with VGG16, we trained classifiers using MobileNetV2 and DenseNet121 on these extracted features. Subsequently, we combined the outputs of both classifiers to form an ensemble model. This ensemble model utilized the VGG16-extracted features to perform the final classification, leveraging the strengths of both MobileNetV2 and DenseNet121 for improved accuracy.

Next Table shows the results of Vgg16 + MobileNetV2 + DenseNet 121 Model on the CicIDS 2017 dataset:

**Table 19:** Results of Vgg16 + MobileNetV2 + Densenet Model on the CicIDS 2017 dataset

| **Metric** | **CicIDS 2017 Dataset** |
|---|---|
| **Accuracy** | 95.24% |
| **Recall** | 92.95% |
| **Precision** | 97.41% |
| **F1-Score** | 95.13% |

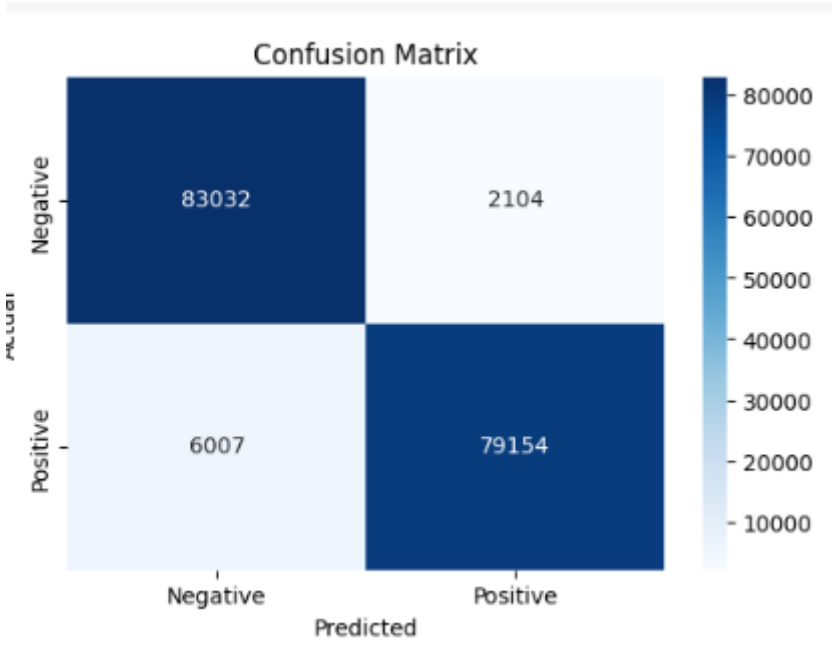



**Figure 36:Confusion matrix of Vgg16 , MobileNetV2 and DenseNet 121 Model on CicIDS2017 Dataset**

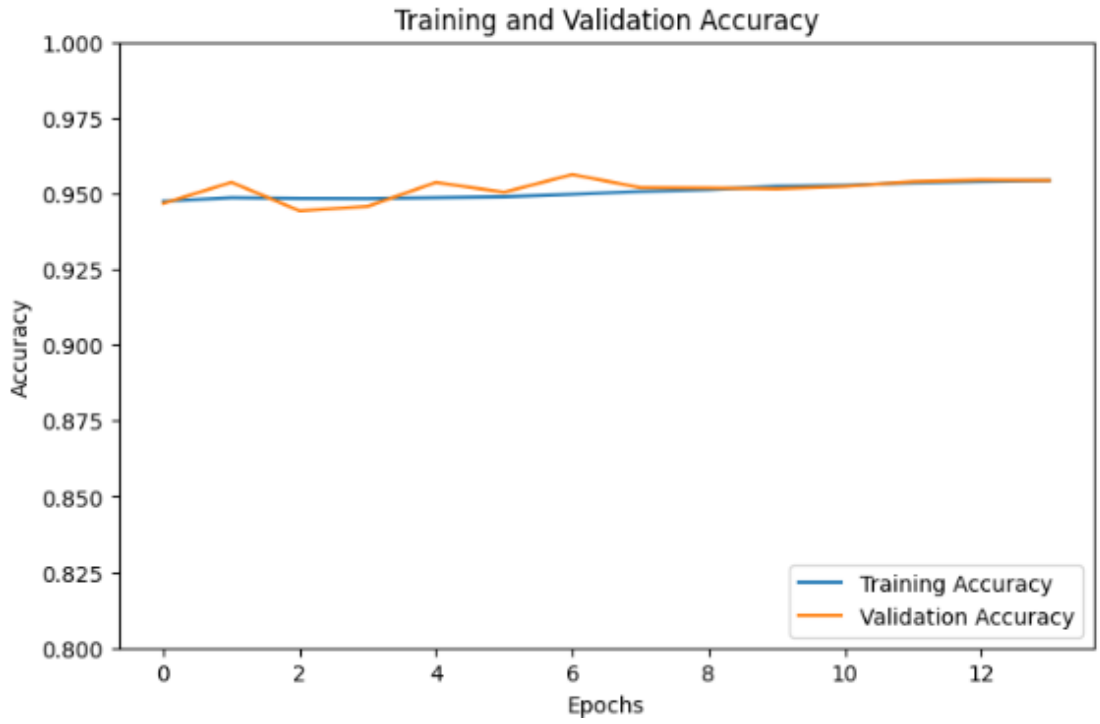



**Figure 37:Train and Testing Accuracy Graph of Vgg16 , MobileNetV2and DenseNet 121 Model on CicIDS2017 Dataset**

## IV.6 Discussion

### IV.6.1 Analyze of results obtained

Next Table summaries all the results of our work:

**Table 20**:Summary of the work results

| | CicIDS2017 Dataset | | | | | CicDDos2019 dataset | |
|---|---|---|---|---|---|---|---|
| | AE+ CNN1 D | **AE+ CNN2D** | AE+ FFDNN | AE+ LSTM | VGG16+ Mobilenet V2+ DenseNet-121 | AE+ CNN1D | **AE+ CNN2D** |
| Accuracy | 97.99% | **98.23%** | 97,68% | 98.01% | 95.24% | 97.69% | **98.43%** |
| Recall | 96.57% | **96.82%** | 95,83% | 98,09% | 92.95% | 97.93% | **99.38%** |
| Precision | 99.39% | **99.63%** | 99,50% | 98.32% | 94.71% | 97.54% | **97.59%** |
| F1score | 97.96% | **98.20%** | 97,63% | 98,20% | 95.13% | 97.74% | **98.48%** |

- AE+CNN 1D model demonstrates robust performance with high precision, that indicates a low rate of false positives which is crucial for effective intrusion detection the accuracy and F1score also suggest a well-balanced model.
- AE+CNN 2D model achieves the highest precision among all the models, making it exceptionally effective at minimizing false positives ,its superior accuracy and F1score position it as the top performer for the CICIDS2017 and CICDDOS 2019 datasets.
- AE+FFDNN model exhibits high precision, its recall is the lowest among the models, indicating a potential to miss more actual intrusions, Although its overall performance remains strong, it is less balanced compared to other models.
- AE+LSTM model offers balanced performance with high recall and precision, demonstrating its effectiveness in training sequential data, its impressive accuracy and F1score further underline its reliability.

- VGG16+ MobileNetV2+ DenseNet 121 model demonstrates an impressive accuracy of 95.24% and a high recall of 92.95%, highlighting its effectiveness in identifying actual intrusions. However, its precision at 97.41% suggests a tendency towards false positives. The F1-Score of 95.13% reflects strong overall performance, yet the model shows a slight imbalance, favoring recall over precision, which might result in more missed intrusions compared to other models.

Comparing all models based on their performance across both datasets, AE+CNN 2D stands out due to its high precision, a key factor in minimizing false positives , given the importance to reducing false positives, making it the most effective at detecting intrusions while maintaining a low rate of false positives.

These deep learning models offer solutions for intrusion detection in IoT networks ensuring effective detection and operational efficiency.

### IV.6.2 Comparison with existing related works

In this study, we have used various deep learning models to create an effective Intrusion Detection System (IDS),the models' performance was evaluated and compared with existing literature to highlight their effectiveness in detecting network intrusions, the models tested include AE+CNN1D, AE+CNN2D, AE+FFDNN, AE+LSTM, and "VGG16 combined with MobileNetV2 and DenseNet 121", Table 20 summarizes the performance metrics of my models existing related works:

**Table 20:** Comparison metrics with existing related works used CicIDS2017 Dataset

| **Model** | **Source** | **Accuracy** | **Recall** | **Precision** | **F1-Score** |
|---|---|---|---|---|---|
| **AE+CNN1D** | Our model | 97.99% | 96.57% | 99.39% | 97.96% |
| | [50]'s Model | 98.40% | 98.50% | 98.30% | 98.40% |
| | [51] 's Model | 96.23% | 96.00% | 95.80% | 95.90% |
| **AE+CNN2D** | My Model | 98.23% | 96.82% | 99.63% | 98.20% |
| | [52] 's Model | 89.82% | 90.16% | 91.81% | 90.98% |
| | [53] 's Model | 99.48% | 99.30% | 99.50% | 99.40% |
| **AE+FFDNN** | Our Model | 97.68% | 95.83% | 99.50% | 97.63% |
| | [20] 's Model | 95.40% | 95.20% | 94.70% | 94.90% |

| | | | | | |
|---|---|---|---|---|---|
| **AE+LSTM** | Our Model | 98.01% | 98.09% | 98.32% | 98.20% |
| | [54] 's Model | 97.16% | 99.10% | 97.41% | 98.25% |
| **VGG16+ MobileNetV2+ DenseNet 121** | Our Model | 95.24% | 92.95% | 97.41% | 95.13% |
| | [55] 's Model | 99.48% | 99.30% | 99.50% | 99.40% |

- AE+CNN1D Model: Our model achieved an accuracy of 97.99% which is near the results of [39] (98.40%) and higher than [51] (96.23%), the recall and precision for our model are also impressive at 96.57% and 99.39%, respectively, indicating a high detection rate and a low false positive rate.

- AE+CNN2D Model: our model's accuracy (98.23%) is higher than [52] (89.82%) but slightly lower than [52] (99.48%), With a recall of 96.82% and a precision of 99.63% that demonstrates good performance especially in precision suggesting it effectively identifies true positive instances.

- AE+FFDNN Model: our model's accuracy of 97.68%, outperforming the model in [52] (95.40%),the precision and recall of my model are notably high, with precision at 99.50%, ensuring that the model is robust in correctly identifying intrusion.

- Our AE+LSTM Model's performance is commendable, with an accuracy of 98.01%, which is slightly better than [52] (97.16%),the recall and precision values are both strong, reinforcing the model's effectiveness in detecting intrusions and reducing false positives

- Our VGG16+MobileNetV2+ DenseNet 121 model achieved an accuracy of 95.24%, which is lower compared to [53]'s model, which attained a high accuracy of 99.48%. While our model exhibits a recall of 92.95%, indicating its strong capability in identifying actual intrusions, its precision is at 97.41% suggesting a low rate of false positives . In contrast, [53]'s model not only surpasses in accuracy but also in precision, with a notable precision of 99.50%. Additionally, our model's F1-Score of 93.58% reflects a balanced performance, yet it falls less compared to the nearly perfect F1-Score of 99.40% achieved by [53]'s model, which demonstrates better performance across all metrics.

The comparative analysis of our models with those from related articles indicates that extracting features with autoencoder using the classification models that we developed

(CNN1D,CNN2D, FFDNN and LSTM) show strong performance in accuracy, recall, and precision, this suggests that they are highly effective for use in an Intrusion Detection System, while the Vgg16 , MobileNetV2 and DenseNet 121 model despite lower accuracy.

These results validate the efficacy of using deep learning models for intrusion detection and highlight the potential improvements over existing models in the literature.

# V CONCLUSION AND FUTURE WORKS

With the expansion of the Internet of Things “IoT”, we are entering an era where devices communicate directly with each other, with a forecast of 30.9 billion active IoT devices by 2025, this 55% growth in the IoT sector attracts interest from both professionals and cybercriminals , the developers focused on building high-performance IoT devices, often neglect security due to limited computing resources because the cyberattacks on IoT are not limited to individual devices but can facilitate access to a multitude of connected services making early detection of intrusions on IoT networks crucial.

This research offers an in-depth study of intrusion detection systems (IDS) using deep learning, building effective deep learning models is increasingly challenging due to data complexity ,despite these challenges, deep learning has shown remarkable learning potential and remains a promising area of research in AI.

Our work presents a model based on Autoencoder “AE”, and various convolutional neural network architectures “CNN-1D, CNN-2D, LSTM, MobileNetV2, VGG16 , DenseNet 121 FFDNN” to enhance the performance and security of IoT networks ,we examined deep learning models for IDS in an IoT network, analyzing traffic to predict possible intrusions and abnormal behavior and the proposed models were trained and tested using the CICIDS2017 and CICDDOS2019 datasets.

Significant research is still needed to advance IoT security and improve threat detection rates ,it is essential to design and implement robust security procedures both inside and outside IoT devices, as IoT networks will be central to our systems ,our future work focuses on developing innovative solutions to analyze encrypted network traffic, identifying characteristics to enhance the detection of suspicious activities while optimizing computing resources we plan to enhance our model by training on the CICIoT2023 dataset, which is a very large dataset, and we aim to implement the "VGG16 MobileNetV2 and DenseNet 121 " model due to its fast detection speed as mentioned in existing works, we are committed to implement our model in Iot devices and applications

# VI Bibliography


[1] B. Noor and S. Qadir, “Machine Learning and Deep Learning Based Model for the Detection of Rootkits Using Memory Analysis,” *Applied Sciences*, vol. 13, no. 19, p. 10730, 2023.

[2] B. Noor and S. Qadir, “Machine Learning and Deep Learning Based Model for the Detection of Rootkits Using Memory Analysis,” *Applied Sciences*, vol. 13, no. 19, p. 10730, 2023.

[3] G. K. Moinudeen, F. Ahmad, D. Kumar, Y. Al-Douri, and S. Ahmad, “IoT applications in future foreseen guided by engineered nanomaterials and printed intelligence technologies a technology review,” *International Journal of Internet of Things*, vol. 6, no. 3, pp. 106–148, 2017.

[4] P. Sethi and S. R. Sarangi, “Internet of things: architectures, protocols, and applications,” *Journal of electrical and computer engineering*, vol. 2017, no. 1, p. 9324035, 2017.

[5] N. W. Khan *et al.*, “A hybrid deep learning-based intrusion detection system for IoT networks,” *Math. Biosci. Eng*, vol. 20, no. 8, pp. 13491–13520, 2023.

[6] H. F. Bel and S. Sabeen, “A survey on IoT security: attacks, challenges and countermeasures,” *Webology*, vol. 19, no. 1, pp. 3741–3763, 2022.

[7] K. Kim, M. E. Aminanto, and H. C. Tanuwidjaja, *Network intrusion detection using deep learning: a feature learning approach*. Springer, 2018.

[8] D. Akgun, S. Hizal, and U. Cavusoglu, “A new DDoS attacks intrusion detection model based on deep learning for cybersecurity,” *Comput Secur*, vol. 118, p. 102748, 2022.

[9] S. Tsimenidis, T. Lagkas, and K. Rantos, “Deep learning in IoT intrusion detection,” *Journal of network and systems management*, vol. 30, no. 1, p. 8, 2022.

[10] P. Vanin *et al.*, “A study of network intrusion detection systems using artificial intelligence/machine learning,” *Applied Sciences*, vol. 12, no. 22, p. 11752, 2022.

[11] K. K. Vaigandla and J. Benita, “Filter Bank Multicarrier for Next Generation Wireless Communications: An Overview,” in *Proceedings of International Conference on Artificial Intelligence, 5G Communications and Network Technologies (ICA5NT’23)*, 2023, pp. 714–725.

[12] A. Abbas, M. A. Khan, S. Latif, M. Ajaz, A. A. Shah, and J. Ahmad, “A new ensemble-based intrusion detection system for internet of things,” *Arab J Sci Eng*, pp. 1–15, 2022.

[13] N. W. Khan *et al.*, “A hybrid deep learning-based intrusion detection system for IoT networks,” *Math. Biosci. Eng*, vol. 20, no. 8, pp. 13491–13520, 2023.

[14] M. M. Taye, “Understanding of machine learning with deep learning: architectures, workflow, applications and future directions,” *Computers*, vol. 12, no. 5, p. 91, 2023.

[15] Y. Otoum, D. Liu, and A. Nayak, “DL-IDS: a deep learning–based intrusion detection framework for securing IoT,” *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 3, p. e3803, 2022.

[16] B. Baligh and M. L. Hamza, “Approche basée sur l’apprentissage profond pour la détection d’intrusion réseau”.

[17] E. Benavides, W. Fuertes, S. Sanchez, and M. Sanchez, “Classification of phishing attack solutions by employing deep learning techniques: A systematic literature review,” *Developments and Advances in Defense and Security: Proceedings of MICRADS 2019*, pp. 51–64, 2020.

[18] M. Macas, C. Wu, and W. Fuertes, “A survey on deep learning for cybersecurity: Progress, challenges, and opportunities,” *Computer Networks*, vol. 212, p. 109032, 2022.

[19] K. Choudhary *et al.*, “Recent advances and applications of deep learning methods in materials science,” *NPJ Comput Mater*, vol. 8, no. 1, p. 59, 2022.

[20] O. Elnakib, E. Shaaban, M. Mahmoud, and K. Emara, “EIDM: Deep learning model for IoT intrusion detection systems,” *J Supercomput*, vol. 79, no. 12, pp. 13241–13261, 2023.

[21] A. C. Mazari, N. Boudoukhani, and A. Djeffal, “BERT-based ensemble learning for multi-aspect hate speech detection,” *Cluster Comput*, vol. 27, no. 1, pp. 325–339, 2024.

[22] A. Alshaibi, M. Al-Ani, A. Al-Azzawi, A. Konev, and A. Shelupanov, “The comparison of cybersecurity datasets,” *Data (Basel)*, vol. 7, no. 2, p. 22, 2022.

[23] P. Sun *et al.*, “DL-IDS: extracting features using CNN-LSTM hybrid network for intrusion detection system,” *Security and communication networks*, vol. 2020, no. 1, p. 8890306, 2020.

[24] D. Agostinello, A. Genovese, and V. Piuri, “Anomaly-based intrusion detection system for DDoS attack with Deep Learning techniques,” in *Proceedings of the 20th International Conference on Security and Cryptography. 1*, SCITEPRESS, 2023, pp. 267–275.

[25] T. Saba, A. Rehman, T. Sadad, H. Kolivand, and S. A. Bahaj, “Anomaly-based intrusion detection system for IoT networks through deep learning model,” *Computers and Electrical Engineering*, vol. 99, p. 107810, 2022.

[26] M. Soltani, M. J. Siavoshani, and A. H. Jahangir, “A content-based deep intrusion detection system,” *Int J Inf Secur*, vol. 21, no. 3, pp. 547–562, 2022.

[27] A. Luo, “Intrusion detection system for internet of vehicles based on ensemble learning and cnn,” in *Journal of Physics: Conference Series*, IOP Publishing, 2022, p. 012014.

[28] J. L. Leevy and T. M. Khoshgoftaar, “A survey and analysis of intrusion detection models based on cse-cic-ids2018 big data,” *J Big Data*, vol. 7, pp. 1–19, 2020.

[29] A. E. Ibor, F. A. Oladeji, O. B. Okunoye, and O. O. Ekabua, “Conceptualisation of Cyberattack prediction with deep learning,” *Cybersecurity*, vol. 3, pp. 1–14, 2020.

[30] M. C. P. Saheb, M. S. Yadav, S. Babu, J. J. Pujari, and J. B. Maddala, “A review of DDoS evaluation dataset: CICDDoS2019 dataset,” in *International Conference on Energy Systems, Drives and Automations*, Springer, 2021, pp. 389–397.

[31] D. Zegarra Rodríguez, O. Daniel Okey, S. S. Maidin, E. Umoren Udo, and J. H. Kleinschmidt, “Attentive transformer deep learning algorithm for intrusion detection on IoT systems using automatic Xplainable feature selection,” *PLoS One*, vol. 18, no. 10, p. e0286652, 2023.

[32] M. Zhong, Y. Zhou, and G. Chen, “Sequential model based intrusion detection system for IoT servers using deep learning methods,” *Sensors*, vol. 21, no. 4, p. 1113, 2021.

[33] A. Gueriani, H. Kheddar, and A. C. Mazari, “Enhancing IoT Security with CNN and LSTM-Based Intrusion Detection Systems,” in *2024 6th International Conference on Pattern Analysis and Intelligent Systems (PAIS)*, IEEE, 2024, pp. 1–7.

[34] E. U. H. Qazi, A. Almorjan, and T. Zia, “A one-dimensional convolutional neural network (1D-CNN) based deep learning system for network intrusion detection,” *Applied Sciences*, vol. 12, no. 16, p. 7986, 2022.

[35] A. Aouatif, B. Omar, and C. Hakima, “An optimized neural network-based IDS against DDoS attacks,” in *2023 10th International Conference on Wireless Networks and Mobile Communications (WINCOM)*, IEEE, 2023, pp. 1–6.

[36] J. Figueiredo, C. Serrão, and A. M. de Almeida, “Deep learning model transposition for network intrusion detection systems,” *Electronics (Basel)*, vol. 12, no. 2, p. 293, 2023.

[37] W. Deabes and K. E. Bouazza, “Residual Autoencoder Deep Neural Network for Electrical Capacitance Tomography.,” *Computers, Materials & Continua*, vol. 73, no. 3, 2022.

[38] A. Badshah *et al.*, “AAKE-BIVT: Anonymous authenticated key exchange scheme for blockchain-enabled internet of vehicles in smart transportation,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 2, pp. 1739–1755, 2022.

[39] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[40] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[41] K. O’shea and R. Nash, “An introduction to convolutional neural networks,” *arXiv preprint arXiv:1511.08458*, 2015.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Adv Neural Inf Process Syst*, vol. 25, 2012.

[43] D. Kilichev, D. Turimov, and W. Kim, “Next–Generation Intrusion Detection for IoT EVCS: Integrating CNN, LSTM, and GRU Models,” *Mathematics*, vol. 12, no. 4, p. 571, 2024.

[44] Y. Jiang, A. Vaicaitis, J. Dooley, and M. Leeser, “Efficient Neural Networks on the Edge with FPGAs by Optimizing an Adaptive Activation Function,” *Sensors*, vol. 24, no. 6, p. 1829, 2024.

[45] H. C. Altunay and Z. Albayrak, “A hybrid CNN+ LSTM-based intrusion detection system for industrial IoT networks,” *Engineering Science and Technology, an International Journal*, vol. 38, p. 101322, 2023.

[46] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.

[47] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, “Mobilenetv2: Inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[48] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely connected convolutional networks,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[49] A. Gueriani, H. Kheddar, and A. C. Mazari, “Deep Reinforcement Learning for Intrusion Detection in IoT: A Survey,” in *2023 2nd International Conference on Electronics, Energy and Measurement (IC2EM)*, IEEE, 2023, pp. 1–7.

[50] I. Ullah, A. Ullah, and M. Sajjad, “Towards a Hybrid Deep Learning Model for Anomalous Activities Detection in Internet of Things Networks. IoT 2021, 2, 428–448.” s Note: MDPI stays neutral with regard to jurisdictional claims in published …, 2021.

[51] M. Catillo, A. Pecchia, and U. Villano, “Successful intrusion detection with a single deep autoencoder: theory and practice,” *Software Quality Journal*, vol. 32, no. 1, pp. 95–123, 2024.

[52] F. Alanazi, K. Jambi, F. Eassa, M. Khemakhem, A. Basuhail, and K. Alsubhi, “Ensemble Deep Learning Models for Mitigating DDoS Attack in Software-Defined Network.,” *Intelligent Automation & Soft Computing*, vol. 33, no. 2, 2022.

[53] R. Kalpana, “Recurrent nonsymmetric deep auto encoder approach for network intrusion detection system,” *Measurement: Sensors*, vol. 24, p. 100527, 2022.

[54] R. A. Disha and S. Waheed, “Performance analysis of machine learning models for intrusion detection system using Gini Impurity-based Weighted Random Forest (GIWRF) feature selection technique,” *Cybersecurity*, vol. 5, no. 1, p. 1, 2022.

[55] H.-C. Lin, P. Wang, K.-M. Chao, W.-H. Lin, and J.-H. Chen, “Using deep learning networks to identify cyber attacks on intrusion detection for in-vehicle networks,” *Electronics (Basel)*, vol. 11, no. 14, p. 2180, 2022.