Thesis Report

A Hybrid Deep Learning Approach for Network Intrusion Detection and Prevention

SUBMITTED BY

Niloy Kumar Joy

Roll No: ASH2011017M

Session: 2019-2020

Year: 04; Term: 02

Department of Information and Communication Engineering

THESIS SUPERVISOR

Dr. Md. Ashikur Rahman Khan

Chairman & Professor

Department of Information and Communication Engineering

Noakhali Science and Technology University

Noakhali-3814, Bangladesh

July 2025

DECLARATION

This thesis report is submitted to the Department of Information and Communication Engineering, Noakhali Science and Technology University, in partial fulfillment of the requirements for having the B.Sc Engineering degree in ICE. This is also needed to certify that the thesis work is under the B.Sc in Engineering Course of department of ICE, NSTU titled “ICE-4218: Project and Thesis”. So, I declare that this thesis report has not been submitted elsewhere for the requirement of any kind of degree, diploma and publication.

Niloy Kumar Joy

Student of Bachelor of Science,

Department of Information and Communication Engineering,

Noakhali Science and Technology University,

Sonapur – 3814, Noakhali, Bangladesh.

ID : ASH2011017M

Session : 2019-20

ACCEPTANCE

The thesis report is submitted to the Department of Information and Communication Engineering, Noakhali Science and Technology University, Noakhali in partial fulfillment of the requirements for having the B.Sc Engineering degree in ICE.

Dr. Ashikur Rahman Khan,

Professor,
Department of Information and Communication Engineering

Noakhali Science and Technology University.

Sonapur – 3814, Noakhali, Bangladesh.

ABSTRACT

With the ever-changing online threats that are entailing fast development, there are increased concerns on strong and intelligent network security. The static rule-based traditional intrusion detection systems (IDS) based on signature-based approach are not able to detect new or advanced attacks. The present thesis suggests a combination of both deep learning technologies based on Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM) networks to improve the detection of intrusion and allow the intrusion prevention proactive measures. By combining the spatial feature extraction capability of CNN and the temporal dependence capture capacity of LSTM, CNN-LSTM model can be said to make best use of the two models when the domain is network traffic. On the NSL-KDD dataset which is a benchmark dataset that comprises of different classes of attacks namely DoS, Probe, R2L, and U2R, the model was trained and tested versus other deep learning models like the MLP, standalone LSTM and autoencoder. The presented model beat others, with the highest accuracy of 95.79 percent, and better precision, recall, and F1-score. Besides detection, this paper discusses rudimentary prevention systems to narrow the response time and constrain attacks. The findings prove that a hybrid deep learning architecture is more likely to offer a more precise, flexible, and integrated answer to intrusion detection and prevention. The study can be used to develop intelligent cyber security solutions and result as the precursor of real-time and self-learning IDS applications in the future.

Keywords: Network Intrusion Detection, CNN-LSTM, Deep Learning, NSL-KDD Dataset, Cybersecurity, Intrusion Prevention, Hybrid Neural Network, Anomaly Detection

TABLE OF CONTENT

LIST OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Introduction

In the highly inter-connected in modern-day digitalization, the over-reliance on the computer network has at the same time exposed people to the dangers of cyber-attacks and authorized intrusion. Network intrusion, which can be considered any unauthorized accessing of information system that intends to peep into the confidentiality, integrity or availability of a system can as well be a critical threat to organizational and national security[1]. As the number of gadgets connected to the Internet of Things (IoT) is increasing exponentially, and considering that a worldwide cybercrime is projected to reach 10.5 trillion dollars per annum by 2025 (Cybersecurity Ventures), the necessity in powerful and intelligent84 security measures is even more relevant now than ever before.

The conventional security systems like the firewalls and signature-based intrusion detection systems (IDS) cannot identify the new or advanced attacks like zero-day exploitation or polymorphic malware[3]. Such limitations require the establishment of intelligent and adaptive systems which are able not only to track down well-known and also unrecognized threats but also assist in reducing the disorders prior to them getting extreme harm[8]. The results have been encouraging towards the implementation of deep learning models particularly hybrid architecture composed of Convolutional Neural Networks (CNNs) conjoined with Long Short-Term Memory(LSTM) networks to serve effectively in acquiring the spatial as well as the temporal

characteristics of the network traffic data, such that, intricate attacks patterns could be identified[26].

In this thesis, a network intrusion detection system (NIDS) based on deep learning is proposed and implemented on the network intrusion detection system that uses a hybrid CNN-LSTM model based on the NSL-KDD dataset as a well-known benchmark in IDS performance measurement. The CNN part extracts the high-level spatial characteristics of the network traffic data, and as a result, the LSTM part captures temporal dependencies, which, in turn, makes the latter part of the model extremely effective in detecting both known and a new attack pattern including DoS, Probe, U2R, and R2L.

Moreover, its study is also directed at the prevention aspect that is not covered when studying intrusion detection in many cases. Taking preventive measures without the mechanism of detecting intrusions will make detection system to be incomplete. Thus we investigate mechanisms and tactics like response mechanism, alert mechanism and update of firewall policy which can be used as action mechanism after the detection of an attack to be made to eliminate further propagation of the attack.

The goal of the result of this dissertation would be to help in the creation of a more secure cyberspace by developing the system which is not only able to detect intrusions with higher accuracy using the advanced deep learning techniques but is also able to facilitate real time or near real time prevention solutions as well. This twofold functionality is needed against the rising challenge on the current networks.

1.2 Problem Statement

In the modern domain of digitalization, the safety of computer networks gained top priorities. Both the amount and sophistication of cyberattacks are on the rise and is presenting a serious concern to organizations, governments as well as to individuals. IDS In traditional intrusion detection systems, they are ineffective any more in counteracting sophisticated as well as cyber threats. DDoS and Smurf attacks, R2L, U2R, Probe, and many more that are prevalent among the

benchmark data sets like NSL-KDD demand smart detection tools which are able to interpret spatial as well as sequential pattern in network activities. In addition to that, identifying an intrusion and failing to engage in preventive measures makes systems susceptible to further attacks. This thesis attempts to deal with these problem by creating a CNN-LSTM-based detection framework with inclusion of prevention measures.

I. Shortcomings of the Older Intrusion Detection System

Signature-based and statistical IDS are common and to great disadvantages. Such systems cannot detect new or unfamiliar attacks (zero-day vulnerabilities) which are not caught by their predetermined patterns. They are also subject to excessive false positive and usually do not generalize to other types of network environments. They consequently fail to provide a source of surety in the dynamic and large-scale networks where the attack vectors keep oscillating. This necessitates the need to consider more intelligent and adaptive strategies, which include deep learning, to empower the detection activity.

II. Why NIDS need Deep Learning

Conventional machine learning models have been promising when used in intrusion detection although they could rarely cover the relationships between the network traffic data. The deep learning models provide such advantages. CNNs are effective in the extraction of spatial features of raw network information. These strengths join together to provide even more precise and robust intrusion detection, which would be able to detect even minor patterns suggesting an attack

III. The Significance of Prevention in the Relation to Detection

Current systems only focus on the intrusion detection trying to ignore the aspect of prevention. Even without the timely mitigation, the significant damage can occur due to the detection. In a case of the DoS attack, catching up on the attack once it has crippled a server is like security taken down the drain. Thus, detection models have to be a part of overall cybersecurity system, which includes the integration of preventive activities like automated response system, access control policies or alert generating systems. Such a duals approach is a guarantee that not only threats are identified early, but their defense is also active.

IV. Relevance of NSL-KDD Dataset

The labeled nature of the NSL-KDD dataset and the variety of represented attack types make it one of the most common benchmarks used in the intrusion detection research. It contains both malicious and normal traffic grouped in four broad groups, DoS, Probe, R2L and U2R. Those attacks are representative of the real world, and, thus, the dataset can be used to train and evaluate deep learning models. Even though serving as an improved version of the older KDD Cup 1999 dataset, NSL-KDD also has certain problematic aspects, including the presence of class imbalance and feature overlaps, which makes it an excellent candidate to test the performance of even sophisticated models such as CNN-LSTM.

1.3 Motivation for This Study

The rationale of the study lies in the fact that cyber threats are becoming more common in the wild and the regular security systems fail to reliably identify and address modern malicious activity against networks. Over the past few years, cyberattacks have not only been increasing in frequency but they have also been more intelligent as they have attacked the critical infrastructures, financial institutions, government agencies and even individual users. The presence of the Internet of Things (IoT) and increased incorporation of digital services have led to a massive rise in the attack surface. Global cybersecurity reports reveal that millions of attempts to intrude are launched each day, and some are unnoticed because of the insufficient capabilities of the existing systems of detection. The older systems of intrusion detection (IDS) that have been using signature-based detection method are not very effective against the attacks that are completely new or the zero-day attacks. Besides, such systems have been seen to produce a high false positive rate and hence they are ineffective when it comes to large scale or real-time scenes. These shortcomings require the implementation of smarter and more agile solutions that will learn and be able to respond to new types of attacks. In the end, it is the possibility of creating an all-inclusive intrusion detection and prevention system to enhance cybersecurity systems, mitigate the possibility of unnoticed hacking attacks, and lead to secure and stronger network systems, which acts as the driving force behind this research.

1.4 Research Objectives

Coupled with the emerging cyber threats and the inadequacies of the conventional methods of intrusion detection, there will always be a tremendous increased demand of intelligent solutions capable of adequately detecting and preventing network attacks. This study will focus on model design and implementation based on the use of a hybrid deep learning technique that makes use of CNN LSTM to enhance the accuracy and reliability of the intrusion detection systems and also includes a combinational approach of preventive mechanisms to minimize the effects of the detected attacks. This study aims at achieving the following objectives:

I. In order to examine the pitfalls of traditional and currently used machine learning-based detection of intrusions and point out a necessity in the usage of deep learning techniques in contemporary networks.

II. To create a hybrid Convolutional Neural Network (CNN) and Long Short-Term Memory (LSTM): model that can detect multiple type of network Attacks with the help of spatial and temporal features extraction of NSL-KDD dataset.

III. To compare what performance of proposed CNN-LSTM can achieve in terms of common criteria, including accuracy, precision, recall, F1-score, and a confusion matrix with other base models, such as traditional MLP, LSTM, and Autoencoder.

IV. To sketched and suggest a prevention structure that has the capacity to re-act to identified intrusions by means of computerized or semi-automated activities, like alert or traffic impediment.

V. To test the character of the proposed system regarding the real-time or near-real-time qualities and archive the challenges of implementing the such illustrations in the performance of dynamic networks.

VI. To contribute to network security as a whole providing the full-scope solution taking into consideration the both intrusion detection and the intrusion prevention (in particular to meet the current and future threats with the cybercriminals).

1.5 Contribution of This Study

The study fits into the emerging evidence-based practice on intelligent cybersecurity since it provides a complete way of identifying and curbing network hack using deep learning. Whereas some of the available works either detect or use only one model, the presented work includes a hybrid method of deep learning and preventive measures to reinforce a network. As discussed below, the contributions are:

I. Training CNN-LSTM Based Intrusion Detection Model

A new hybrid deep learning framework which integrates Convolutional Neural Networks (CNN) to seek spatial feature with Long Short-Term Memory (LSTM) networks to seek temporal dependency is proposed to classify sequences. The NSL-KDD dataset is used in training and testing this model which is comprised of different categories of attacks such as DoS, Probe, R2L, and U2R.

II. Better Detection precision of Multiple Assault Types

The suggested CNN-LSTM system is considered to work better when it comes to locating complex or multi-level attacks and provides more accuracy and fewer false-alarms than typical techniques. The outcomes depict better findings especially in the identification of rare type attacks like U2R and R2L.

III. Prevention Mechanisms Integration

Such a research contrasts with many other available studies, which have been focusing on prevention in addition to detection. It suggests a simple framework to react on detected intrusions by using automated messages, blocking of access and making appropriate policy changes, which makes part of the proactive defense mechanism.

IV. testing and analysis on NSL-KDD Data set

The dataset used in the study is NSL-KDD which has been used as a benchmark to conduct training/testing to have a standard evaluation and compared with other models. Model performance improvements are also done through data preprocessing, selection of features and balancing classes.

V. How to Deal with Real World Security.

The current work focuses on the aspects of being applicable to real life with the focus on such essential issues as zero-day attack detection, control of false positives, and the dynamics in environments in which IDS is applied. The suggested method is going toward the real-time or near real-time intrusion remedy.

VI. Research contribution Bed around Deep Leaning Hybridwater in NIDS

This paper adds to the scholarly research as it helps to prove the efficiency of CNN-LSTM architecture providing us with the possibility to move further and create something better and more useful.

1.6 Scope of This Study

This study investigates ways of designing, developing, and testing a hybrid deep learning architecture-based network intrusion detection and prevention system with CNN LSTM architecture. It is aimed at the analysis of network traffic data of NSL-KDD dataset to recognise different kinds of network attacks such as DoS, Probe, R2L and U2R. This paper also handles the detection accuracy by using the spatial and temporal characteristics of the data to enhance

detection and also handles more than detection through integration of some simple prevention techniques like using automated warnings or grants. Although the study is restricted to the domain of supervised learning and benchmark dataset but establishes a foundation of the future improvement of this paper to include the deployment of the model in real-time deployment, application of modern datasets, as well as integration into real networks, to actively mitigate threats that give rise to the threats.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

The introduction of machine and deep learning methods has improved the sphere of network intrusion detection dramatically. This literature review has been well prepared to give a good account of the most powerful and recent contributions in this field. It embraces a broad area of research which considers many different datasets which include NSL-KDD, UNSW-NB15, CICIDS2017, KDD99 among other datasets and a plethora of models that involves Autoencoders, Convolutional Neural Networks(CNN), Recurrent Neural Networks(RNN), Long Short-Term Memory(LSTM), Random Forests, and a constellation of hybrid ensemble models. All the studies have been examined with respect to features utilized, techniques employed, precision obtained, alongside its given benefits and shortcoming. Not only has this review identified the performance standards of various models, but it has also given a clue to the updating challenges and research lapses in the intrusion detection systems. It is used as a guiding source of reference upon which the models that will be used in the successive chapters of this thesis will be developed and justified.

Song et al. (2021) applied the deep autoencoder to create an intrusion detector within a network using 41 standard features used by the NSL-KDD dataset in 2021. Unsupervised feature learning technique was used by them where the model learns significant patterns in the data supplied automatically. The system was found out to have an accuracy of 89.5 percent which signifies that the performance was moderately successful in terms of recognition of different network attacks. One of the strengths of this approach is that it does not require labeled data to learn, something that makes it applicable in real-life settings, where labeled data is limited. The model has however shown weaknesses in the detection of certain classes of attacks, probably because of the generality problems of the autoencoder structure in training with very diverse sets of attacks[1].

Ghani et al., in 2023, introduced Deep Neural Network (DNN) model with much-lesser set of features between 9 to 11 features on both NSL-KDD and UNSW-NB15 datasets. They aimed at making the input space less complicated and computationally less intensive without compromising in the accuracy efficiency much. Its accuracy on UNSW-NB15 and NSL-KDD data were 91.29 and 89.03 percent, respectively, which is good. The approach showed that thoughtful feature selection was capable of producing efficient-but-accurate models, which is especially relevant in limited resource settings. Nevertheless, its effectiveness varied as the used feature set was much more important, which implies that it may not cross-generalize to other attack scenarios without re-training[2].

In 2024, Adeola Ajagbe et al. tried the application of 49 flow features obtained through Argus/Bro in the UNSW-NB15 correlation to test a variety of conventional machine learning classifiers, such as Support Vector Machines (SVM), Logistic Regression, Random Forest (RF), Decision Tree and XGBoost. Random Forest was superior among them in terms of robustness and the ability to process imbalanced datasets, and its accuracy reached nearly 90%. The modelers referred to the fact that classical classifiers could be used even in the situation when feature engineering was properly used. Nonetheless, the research was restricted to binary classification hence limiting use of the same in more informative intrusion detection modules that required multiclass classification[3].

Meliboev et al. (2022) used all 43 flow-based features, encoded as one-hot vectors, based on such datasets as UNSW-NB15, KDDCup99, and NSL-KDD. They applied and compared deep learning those being CNN, LSTM, GRU, RNN, and a hybrid CNN+LSTM. KDDCup99 was the best with a 92.3 percent accuracy compared to 78.8 percent that the model has been able to accomplish on NSL-KDD. This difference in the performances underscores the role of the nature of the datasets in increasing the effectiveness of the models. Their research also demonstrated the potential of the permutation of spatial and sequential learning architectures, although not, according to them, all models generalized so well across datasets[4].

In 2022 a Variational Autoencoder (VAE) based feature extraction and Deep Neural Network (DNN) based classifier was introduced on the NSL-KDD dataset by Khanam and co-authors.

The model showed an overall accuracy of 88.08%, and this proves that generative augmentation is efficiently used in enhancing data representation. VAE was able to learn compressed but informative latent spaces and allowed them to excite the power of detection in the DNN. Despite the aspiration, the accuracy of the method failed to get close to the superior levels observed in other studies, which indicates mediocre performance and the opportunity to enhance the approach in practice[5].

According to a study conducted by Shone et al.(2018), a potent intrusion detection framework that is unsupervised by using an Autoencoder-learned feature on the NSL-KDD dataset was developed. They have used Stacked Denoising Autoencoder comprising a Random Forest classifier. The result of this hybrid approach was an impressive accuracy of 97.85% which is a high standard to achieve in intrusion detection by employing deep learning. The method was extremely useful in learning non-linear representations given raw-data. yet, the architecture was complicated in general and primarily the superimposed layers of autoencoders made it impractical to put to work as lightweight apps or in real-time systems[6].

Abdelaziz et al. adopted a Random Forest classifier on 26 critical features obtained via the permutation importance of the CICIDS2017 dataset. In this paper, special attention was paid to the design of the highly interpretable intrusion detection system, which remained operationally high, with the accuracy of 93.31%. Random Forest was so simple and clear that it could be used in explainable AI in security environments such as military. However, the study has lesser macro F1 scores that are as a result of class imbalance which suggests that the study has decreased effectiveness in finding classes that are minority, and that is very common in real-world data sets[7].

Jing and Chen (2019) used the standard features extracted on the UNSW-NB15 dataset and operated Support Vector Machine (SVM) algorithms. In their model, they performed robustly in binary classification experiments having an accuracy of 85.99% whereas under multiclass scenarios, performance plunged to 75.77%. SVM showed its power in processing non-linear data distribution, particularly, in case of simple binary decisions. The overall difference of the performance of binary and multiclass classification however showed a limitation of the

scalability of the SVM approach and this approach would require some more fine-tuning or modification to be used more widely in intrusion detection systems of a complex nature[8].

Elsaid et al. introduced an approach of hybrid deep learning in 2024 that combines GRU and LSTM with Grey Wolf Optimization (GWO) in the networks intrusion type to improve the detection of the intrusion. They used the NSL-KDD and UNSW-NB15 datasets in their research and NSL-KDD and UNSW-NB15 datasets to select the best features with the help of GWO algorithm. The GRU-GWO and LSTM-GWO models recorded excellent binary classification accuracy of 94 percent and 92 percent respectively in the NSL-KDD data set and 90 percent and 79 percent in the UNSW-NB15. The effective approach allows reducing dimensionality of the input feature space with no degradation in the performance of the classification. Nonetheless, there did exist a significant training complexity and computational overhead to GWO tuning, which may impose a limitation on the scalability of such a system within a real-time environment[9].

In 2023, Gou et al. suggested a stacked ensemble learning method in the intrusion detection on the CICIDS2017 database. To balance the distributions, they utilized a mixture of SMOTE and under-sampling, and to obtain such an impressive results, they used a robust ensemble of classifiers such as Random Forest, Decision Tree, K-Nearest Neighbor, XGBoost, and Logistic Regression. The method has demonstrated certain degrees of strength when dealing with complex patterns of intrusion as it obtained a robust multiclass classification accuracy of 96.5%. The use of the ensemble technique increases overall performances of detection since individual classifiers may have some advantageous features. Its greatest weakness is however its high computation cost that could be a challenge during implementation in resource-limited scenarios[10].

In 2024, Asry et al. proposed a low-resource system of network intrusion detection that has shown both good results and low network capacity: a combination of Paragraph Vector Diamond Memory (PV-DM), or so-called Doc2Vec, and the XGBoost classifier. With four preselected features, the model performed with an outstanding accuracy level of 98.92 percent on NSL-KDD dataset, and 82.86 percent on UNSW-NB15. Such limited feature set dramatically cuts down

training and inference time and yet outperforms the 97% accuracy threshold on NSL-KDD. The model can be of benefit especially in classification of binary where the dimensions of the input are small. Nonetheless, the results on UNSW-NB15 were relatively weaker, and this fact shows that the tool might not be broadly generalizable across a wide range of dataset[11].

In 2023, M. Alhanaya and K. Al-Shqeerat introduced the combination intrusion detection model that was improved by Principal Component Analysis (PCA) as a dimensional reduction method. By using KDD_99 and NSL-KDD datasets, the authors made ensemble of ExtraTrees, KNN and Random Forest classifiers, the accuracy of which reached the unbelievable level of 99.89 against the NSL-KDD, with only 30 principal components. The PCA component minimized computation work not only by making model better but also by enhancing model generalization. This work is an illustration of the success of using ensemble learning in combination with data preprocessing to boost performance. Nevertheless, before PCA, its high dimension remains as a limitation, as well as the possible loss of interpretable items in the principal components[12].

S. in 2024 achieved a 2 2 per cent gain. More and colleagues specialized on UNSW-NB15 data set and applied correlation-based feature pruning and exploratory data analysis on enhancing the complete task performance of usual machine learning models. The models were Logistic Regression, SVM, Decision Tree and Rando Forest. The resulting system realised a high test accuracy of 98.63% which was made possible by thorough data preparation and statistically based feature selection. Their method also showed that more simplistic machine learning methods, suitable optimized, could perform as well as more sophisticated deep learning models. Nevertheless, the experiment only considered a single dataset that limits the inferences that can be made about the cross-dataset generalization[13].

In 2020, Wu et al. have managed to come up with a rather peculiar approach that incorporates deep autoencoders, as well as the Random Fourier Features (RFF) and Linear SVM classifier to enhance network intrusion detection. The experiment performed it on the NSL-KDD dataset and became 85.8 percent accurate, which proves that dimensionality reduction and usage of kernel-based classification can be combined. It is a hybrid model that benefits of unsupervised

feature extraction and allows a fast linear classification within the transformed space of features. Although the model may be considered innovative in terms of design, its precision was mediocre in comparison to newer methods indicating that additional optimization could be applied, at least with regards to the multiclass detection detectors[14].

Dinh-Hau Tran and Minho Park, 2024: Dinh-Hau Tran and Minho Park, 2024 came up with an adaptive feature selection strategy that incorporates Random Forests with a Sequential Forward Selection (SFS) algorithm. This method applied to NSL-KDD and UNSW-NB15 datasets resulted in the exceptionally high accuracies of 99.3% and 97.5 respectively. The model achieved a performance guarantee by adapting through normalization and selecting the most pertinent features, thus, improving the efficiency of learning. This fact is supported by the intensity of the ensemble learning and adaptive selection which are proved by high classification rates. The approach however leads to a complex feature selection process needing close tuning and a longer processing time[15].

In 2024, W. H. Aljuaid and others proposed a sophisticated architecture of Convolutional Neural Network that was based on LeNet type, but which was designed respectively to intrusion detection in clouds, properly supporting large volumes. Based on the datasets, e.g., NSL-KDD and CSE-CICIDS2018, the model achieved higher than 98.67% accurate results, showing scalability and goal achievement in real-time cloud exposure. The architecture is well-tailored to identify abnormalities in cloud traffic; therefore, it is very useful in contemporary cybersecurity architecture. However, its specific tailoring to the cloud and CICIDS2018 data might be the factor that restricts its performance on less specific or generalized data[16].

S. Gautam and colleagues in 2022 used such architecture with Bidirectional RNN architecture, including LSTM and GRU blocks and correlation-based feature selection. They tested their model on a CICIDS2017 dataset, with the results reaching 99.13 percent in terms of accuracy. This doth have the advantage of being able to perform sequential learning using bidirectional networks and of minimizing irrelevant dimensions of input. A parameter half of the initial characteristics is enough to make the model less complex and still accurate. However, the

problem is that the model maintains heavy computational needs because of iterative layers, which makes it impractical in large scale or real time systems[17].

In 2022, Thirimanne et al., presented a deep neural network, which detects intrusion attacks in real-time, with a pipeline of encoding and scaling of data. They used 28 characteristics of NSL-KDD dataset to achieve a general accuracy of 81%. The method featured a strong focus on real-time detection accuracy which implies that it may be used in on-line systems. Nevertheless, its overall moderate accuracy implies that it is limited to various pattern of intrusion, particularly, in multiclass scenarios[18].

Wu et al. (2022) recently addressed the class imbalance in the NSL-KDD dataset using, among others, a hybrid SMOTE approach, which combines ADASYN and K-means clustering. Their model adopted a superior Random Forest classifier and demonstrated a low test accuracy of 78.47 %. Whereas the hybrid sampling helped the minority classes to balance against each other during training, the accuracy did not meet those of new models. This indicates how hard it can be to make sure that synthetic data generation can adhere to realistic patterns of attacks. Consequently, the poor generalizability to the test data is an issue of concern[19].

In 2022, Balyan and co-authors have suggested Particle Swarm Optimization (PSO) in improving intrusion detection with the improvement of the parameters of the Random Forest classifier. They tested their ensemble model on the NSL-KDD that achieved an accuracy of 88.15 percent on the five-class classification model. The evolutionary methodology was used to optimize hyperparameters and make it be more robust. But this tuning operation added some new complexity and computational overheads that cannot be always applicable in time-sensitive systems[20].

Cao et al. (2022) used a hybrid deep learning approach that consisted of CNNs and a Pearson correlation-based feature selection procedure in 2022. This was tested in NSL-KDD, UNSW-NB15 and CICIDS 2017 datasets and displayed up to 99.69 percentage accuracy on NSL-KDD and 99.65 percentage on CICIDS2017 however only yielded 86.25 percentage on

UNSW-NB15. Their method derives its strength in the spatial feature extraction capacity to enable it to handle imbalanced data. However, poor performance on UNSW-NB15 demonstrates that the model lacks cross dataset flexibility[21].

In 2022, Le et al. introduced a 10-layer CNN augmented using a Conditional GAN in a minority-class. When tested on the UNSW-NB15 and CICIDS2017 datasets, they produced an accuracy of 96.69 percent and 95.92 percent respectively. The GAN component really improved the training over the underrepresented classes. Nevertheless, minority classes detection accuracy remained quite low on testing sets, indicating that the effectiveness of synthetic samples and their capacity to be generalized to real-life trends is uncertain[22].

In Pak and Han (2023) a two stage LSTM based intrusion detection system is proposed which uses both the packet and session based characteristics (features). They used two benchmark datasets and their model attained accuracies of 95.16 percent and 99.70 percent. Temporal dependencies can be well captured in the use of the sequential architecture as it makes use of the whole communication sessions. The classification speed of the system is slower compared with other systems that may be a factor that hinders the deployment of the systems in real-time, even though the system is accurate[23].

Fu et al. presented in 2022 a complex hybrid framework based on CNNs, channel attention, BiLSTM and oversampling based on synthetic data by ADASYN. The model attained accuracy of 90.73% on NSL-KDD dataset. The two-fold feature of this model is its ability to deal with class imbalance and the extraction of just not spatial but also temporal features. Nevertheless, the rather middle-of-the-road accuracy and complexity gives some idea that it could be most productive in research rather than actual applications[24].

Intrusion detection was developed based on a deep model, which included Stacked Autoencoders, an attention mechanism, and a Multilayer Perceptron according to Tang et al., 2020. Based on NSL-KDD dataset, the model is accurate in its binary classification of 87.74% and multiclass classification of 82.14 percent. The architecture has the ability to learn

hierarchical features effectively, yet its performances are rather low on multiclass problems limiting its practical efficacies[25].

Meliboev et al. (2022), applied sequential raw packet flows of NSL-KDD and UNSW-NB15 dataset and employed several different deep learning models, namely CNN, LSTM, GRU, and RNN in 2022. They achieved the best accuracy of 91.2 percent on UNSW-NB15 demonstrating the advantage of capturing temporal traffic patterns. However, the performance remained influenced by the imbalance in datasets, and this depicted the difficulties of actual deployment[26].

In 2022, Imrana et al. used a 2022 2-D Representation of Bidirectional LSTM with a chi-square feature Selection method on the NSL-KDD dataset. Having only 16 statistical features, the model achieved the result of 95.62% accuracy, which proves the effectiveness of the input reduction without decreasing the level of classification. Nonetheless, overfitting to the known classes of attacks was noted, indicating the lack of flexibility[27].

B. Cao and co-authors suggested half-baked deep model to use ADASYN to balance the classes, RENN to remove the noise, and CNNGRU architecture. They have tested their proposed solution on the NSL-KDD and the CICIDS2017 and the result performance shows accuracies of 99.69% and 99.65 respectively which is rather high. The design is highly effective and brings together the spatial-temporal learning with balanced training. Nevertheless, only in-range measures were reported and no results regarding performance on the unobserved attack types were made[28].

C. Tang and coworkers developed a deep model in which Stacked Autoencoders were used to compress the features and a DNN classifier was used in 2020. On the KDD NSL data, it yielded a 87.74 and 82.14 percent binary and multiclass respectively. Although it excelled relative to classical ML baselines, the NSL-KDD-based analytic tool does not allow generalizing[29].

Most recently, simply a fully connected neural network A. A. Ghani et al. (2023) with relatively small feature vectors nine in the case of UNSW-NB15 managed to obtain 91.29% and 89.03% accuracy on UNSW-NB15 and NSL-KDD respectively. Its computational cost becomes negligible due to the drop in the input and this makes the model able to run in real-time systems. Nevertheless, the multiclass score of the model on NSL-KDD data was not as great which indicates the sacrifice of accuracy to efficiency[30].

2.2 Previous Work at a Glance

Table 2.1 Previous Work at a Glance

Reference	Author's Name	Features Used	Datasets	Methods	Results & Accuracy	Advantages	Limitations
[1]	Song *et al* *(2021)*	All 41 standard NSL-KDD features	NSL-KDD	Deep Autoencoder	89.5%	Unsupervised feature learning	May miss some attack classes
[2]	Ghani *et al (2023).*	Reduced set (9–11 features) per dataset	UNSW-NB15, NSL-KDD	Deep Neural Network (feedforward DNN)	Accuracy 91.29% (UNSW), 89.03% (NSL-KDD)	Low-dimensional input reduces complexity	Relies on selected features
[3]	Adeola Ajagbe *et al (2024)*	49 flow features (Argus/Bro)	UNSW-NB15	SVM, Logistic Reg, RF, Decision Tree, XGBoost	90%	RF robust on imbalanced data	Binary classification only
[4]	Meliboev *et al. (2022)*	43 flow features treated as image (one-hot)	UNSW-NB15, KDDCup 99, NSL-KDD	CNN, LSTM, GRU, RNN, CNN+LSTM	92.3% (KDDCup), 78.8% (NSL)	Comprehensive multi-dataset study	Lower NSL-KDD accuracy
[5]	Khanam *et al. (2022)*	Latent features from VAE	NSL-KDD	Variational Autoencoder + DNN	88.08%	generative augmentation	Moderate accuracy

[6]	Shone *et al. (2018)*	Autoencoder-learned features (unsupervised)	NSL-KDD	Stacked Denoising Autoencoder + RF	97.85%	Very high accuracy	High complexity;
[7]	Abdelaziz *et al*	26 key features (permutation importance)	CICIDS2017	Random Forest	93.31%	Highly interpretable	Macro F1 lower due to class imbalance
[8]	Jing & Chen (2019)	Standard UNSW-NB15 features	UNSW-NB15	Support Vector Machin	Binary accuracy 85.99%(multiclass 75.77%)	SVM handles nonlinear separability	Lower multiclass accuracy
[9]	Elsaid *et al. (2024)*	GWO-selected features (from 49)	NSL-KDD, UNSW-NB15	GRU/LSTM + Grey Wolf Optimization	GRU-GWO: 94% (anomaly), 92% (sig.) on NSL; LSTM-GWO: 94%, 92% UNSW-GRU: 90%,79%; LSTM:93%,79%	Reduces feature dimensionality	Heavy training overhead
[10]	Gou *et al. (2023)*	49 features (SMOTE +Under-sampling)	CICIDS2017	Stacked ensemble (RF, DT, KNN, XGB + LR)	96.5%	Strong multiclass performance via stacking	High computational cost
[11]	Asry *et*		NSL-KD	PV-DM	98.92%	Very high	Under 97

	al. (2024)	4 selected features	D, UNSW-NB15	(Doc2Vec) + XGBoost	(NSL-KDD), 82.86% (UNSW)	NSL-KDD accuracy;	region goal exceeded;
[12]	M. Alhanaya, K. Al-Shqeerat (2023)	PCA dimensionality reduction	KDD'99, NSL-KDD	Ensemble of ExtraTrees, KNN, RF + PCA	99.89% accuracy (NSL-KDD with 30 PCs)	Dramatic accuracy boost via PCA+ensemble	Very high dimensionality
[13]	S. More *et al. (2024)*	Correlation analysis + feature pruning	UNSW-NB15	Standard ML (LR, SVM, DT, RF) + EDA	98.63%	High accuracy; extensive data analysis	Only evaluated on UNSW-NB15
[14]	Y. Wu *et al. (2020)*	Stacked autoencoder; Random Fourier features	NSL-KDD	Deep Autoencoder + RFF kernel + Linear SVM	85.8%	Combines dimensionality reduction with kernel trick	Moderate accuracy
[15]	Dinh-Hau Tran & Minho Park (2024)	RF + Adaptive SFS feature selection	NSL-KDD, UNSW-NB15	Random Forest + Adaptive Normalized SFS	99.3% accuracy (NSL-KDD), 97.5% (UNSW-NB15)	high accuracy	Very complex FS
[16]	W. H. Aljuaid *et al.* (2024)	Advanced CNN (LeNet variant)	NSL-KDD, CSE-CIC IDS2018	Convolutional Neural Network	>98.67% accuracy	Scalable to cloud data	Only cloud/CICIDS2018 context

[17]	S. Gautam *et al. (2022)*	Bidirectional RNN (LSTM/GRU); correlation-based FS	CICIDS2017	Bidirectional RNN + feature selection	99.13%	Uses half of features	Model complexity;
[18]	Thirimanne *et al.* (2022)	28 features of NSL-KDD after encoding	NSL-KDD	Deep Neural Network (DNN) with data encoding and scaling pipeline	81% accuracy	Real-time IDS with high precision	Moderate overall accuracy
[19]	Wu *et al.* (2022)	Hybrid SMOTE (ADASYN+K-means)	NSL-KDD	Enhanced Random Forest (RF) with SMOTE sampling	78.47%	Addresses class imbalance	Low test accuracy
[20]	Balyan *et al.* (2022)	Evolutionary PSO to tune Random Forest parameters	NSL-KDD	Ensemble: Random Forest (RF)	88.15% (5-class multi)	Optimizes RF hyperparameters	complexity from PSO tuning
[21]	Cao *et al.* (2022)	RF + Pearson correlation for feature selection	NSL-KDD, UNSW-NB15, CICIDS2017	Hybrid DL: CNN	86.25% (UNSW-NB15), 99.69% (NSL-KDD), 99.65% (CICIDS)	Addresses imbalance	Lower performance on UNSW-NB15

[22]	Le *et al.* (2022)	10-layer CNN; Conditional GAN for minority-class generation	UNSW-NB15, CICIDS2017	CNN-based DL with data augmentation (GAN)	96.69% (UNSW-NB15), 95.92% (CICIDS 2017)	High accuracy	Poor minority-class detection
[23]	Han & Pak (2023)	Full packet + session features	Two benchmark corpora	Two-stage LSTM: per-packet	95.16% and 99.70% on two data sets	Captures full-session context;	Slower classification speed
[24]	Fu *et al.* (2022)	CNN feature extraction + channel attention	NSL-KDD	Hybrid DL: CNN + attention + BiLSTM + Synthetic minority oversampling (ADASYN) + Stacked AE	90.73%	Handles imbalance	Moderate accuracy (90.7%)
[25]	Tang *et al.* (2020)	Stacked Autoencoder (SAE) + channel attention + DNN	NSL-KDD	Deep model: SAE → Attention → Multilayer Perceptron	87.74% (binary), 82.14% (multi-class)	Learns hierarchical features	lower performance on multi-class
[26]	K. Meliboev, T. Lee,	Sequential network	NSL-KDD, UNSW-	CNN, LSTM, GRU,	91.2%	highest accuracy on	Imbalanced data still

	K. Lee, D. Kim (2022)	traffic (raw packet flows)	NB15	RNN (		UNSW-NB15 (91.2%)	affects performance
[27]	I. Imrana, Y. Xiang, L. Ali, Z. Abdul-Rauf, Y.-C. Hu, S. Kadry, S. Lim (2022)	16 statistical features selected by χ^2 feature ranking (from original NSL-KDD features).	NSL-KDD	χ^2-based feature selection + Bidirectional LSTM	95.62% accuracy	reduces input size while maintaining information.	overfitting to known classes.
[28]	B. Cao, J. Zhang, L. Xie, Y. Qin, C. Chen (2022)	ADASYN oversampling + RENN cleaning for class balancing;	NSL-KDD, UNSW-NB15, CIC-IDS 2017	Convolutional Neural Network + Gated Recurrent Unit hybrid model	(NSL-KDD: 99.69%; CIC-IDS 2017: 99.65%, both above 97%)	captures spatial and temporal features jointly.	Only one reported in-range metric
[29]	C. Tang, N. Luktarhan, Y. Zhao (2020)	Categorical features one-hot encoded + continuous features normalized	NSL-KDD	Stacked Autoencoder (SA) for feature compression + Deep Neural Network classifier (DNN)	(binary): 87.74% accuracy. (Multi-class: 82.14%.)	Outperforms traditional ML baselines	Only evaluated on NSL-KDD.

[30]	A.A. Ghani, H. Virdee, H. Salekzamankhani (2023)	Reduced feature vectors: 9 features for UNSW-NB15	NSL-KDD, UNSW-NB15	Simple fully-connected Neural Network (FFNN) classifier	91.29% accuracy; NSL-KDD (binary): 89.03%	Reduces computational cost with compact model.	Performance on multi-class NSL-KDD was lower

2.3 Summary

The table of a comparative literature review gives a broad overview of current and impactful works in the network intrusion detection systems (NIDS) and reveals numerous approaches to machine learning and deep learning used on famous datasets, including NSL-KDD, UNSW-NB15, CICIDS2017 and KDD-99. Analysis of the reviewed papers shows that there is a definite inclination towards such model type as a hybrid, ensemble type of learning, and feature optimization methods like PCA, GWO, and SMOTE to increase detection rates and manage the issue of class imbalance. The accuracy when using different models varies depending on the model and can be classified as moderate (approximately 78 percent), high (over 90 percent), and exceptionally high (over 99 percent). Autoencoders, CNNs, LSTMs, and Random Forests have been proven to be the most common since they can depict complicated patterns in network traffic. Nevertheless, there are still constraints, such as expensive computation time, poor results in multiclass settings, susceptibility to the feature selection, and difficulties concerning generalization in distinct datasets. On the whole, the table demonstrates the slowly increasing sophistication of the research on NIDS, with the way results are balanced between invention, utility, and the undying requirement of finding appropriate models that are precise and expansive.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In the present thesis, a hybrid approach to network intrusions that uses deep learning techniques is presented, that renders the advantages of both a Convolutional Neural Networks (CNN) and Bidirectional Long Short-Term Memory (Bi-LSTM) and applies them to detect and prevent intrusions on the network. The approach is applied to the NSL-KDD dataset that is preprocessed with label encoding, feature selection based on mutual information, and balancing due to class imbalance by five classes that are DoS, Probe, R2L, U2R, Normal. The raw data is reformatted to fit the CNN-LSTM model and the model is trained with classes having weights, so that majority attacks could be detected easier. accuracy, precision, recall, F1-score, and confusion matrix are used to do the evaluation. It also creates a simulation environment to simulate real-time detection and prevention by generating synthesized Snort alerts, firewall logs, etc in which relevant response against identified threats is executed such as IP blacklist or session closure. This combined solution is to surpass the traditional detection at an active modelling of response techniques, providing verification of the prevention capabilities as well as classification efficiency. The complete approach guarantees a successful and feasible intrusion detection infrastructure that focuses on the precision of identification in addition to the responsiveness during the presence of real-time in a network security environment.

3.2 Model development

The model development, as applied in this study, would be establishing a hybrid deep learning network, which would incorporate Convolution Neural Networks (CNN) and

Bidirectional Long Short-Term Memory (Bi-LSTM) layers, to distinguish and identify network intrusion successfully. The CNN layers will extract the local spatial features in the input data whereas the Bi-LSTM layers will extraction the temporal dependencies and sequence patterns in the network traffic data set. The architecture that forms the model consists of several stacked Conv1D layers with the ReLU activator and the dimensionality reduction through the max-pooling, and afterward, the Bi-LSTM layers that target the better learning of the complex outcomes of the attacks. There are dropout layers applied in between to make sure that overfitting is avoided and finally applies dense layers with softmax activation that results in a five-way classification of DoS, Probe, R2L, U2R, and Normal. The compilation of the model consists of categorical cross-entropy loss and Adam optimizer with callbacks like early stopping callback, learning-rate reduction and model checkpointing to achieve maximum performance in training. It is an architecture aimed at processing remodeled, selected features of the NSL-KDD dataset and trains on the weights of the classes aimed at equalizing the imbalance in the data and eventual goal of attaining a high degree of success and strength in the intrusion identification activity.

3.2.1 Neural Network Models

A type of machine learning model is the neural network, which takes its structure from the brain of a human and can identify complex relationship and patterns in data. They are comprised of weighted interconnected sets of artificial neurons, which change input data by use of activation operations. Some of the best known neural network structures are Long Short-Term Memory (LSTM) and Convolutional Neural Networks (CNNs). CNNs are very good at spatial features extraction while LSTMs are particularly at sequential temporal dependency capture. When used in hybrid models (such as CNN-LSTM models), they can combine space and time knowledge, and hence their effectiveness in task, where CNN-LSTM-based models are superior to other models such as network intrusion detection. Learning of automatized complex patterns related to various forms of cyberattacks is also possible with these models, which minimizes the need to perform manual feature engineering. Neural networks in particular are well suited to dynamic and changing network environments since they are easily adaptable and are precise when working in high-dimensional, imbalanced datasets; early identification and prevention of intrusion are of paramount importance in these types of environments. The four neural networks

models, which were used in the study, include conventional and customized models, as well as, four neural networks models.

a) Multi-Layer Perceptron

The overall formation of a Multi-Layer Perceptron (MLP) includes, broadly, three categories of layers and they are an input layer, one or more hidden layers and, a final output layer. The layers are connected in full i.e. every neuron of one layer is connected with every neuron of the other layer. Raw data is fed into the input layer then fed to the hidden layers where a weight is attached to it and an activation function (ReLU or sigmoid) added so as to introduce non-linearity. The quantity of hidden layer and number of neurons in each layer might depend on the complexity of task. At last, the prediction is followed in the output layer that is usually a softmax activation to make a classification or linear activation during regression. MLPs are optimized by applying gradient descent and backpropagation to minimize a loss e.g. mean squared error or categorical crossentropy.

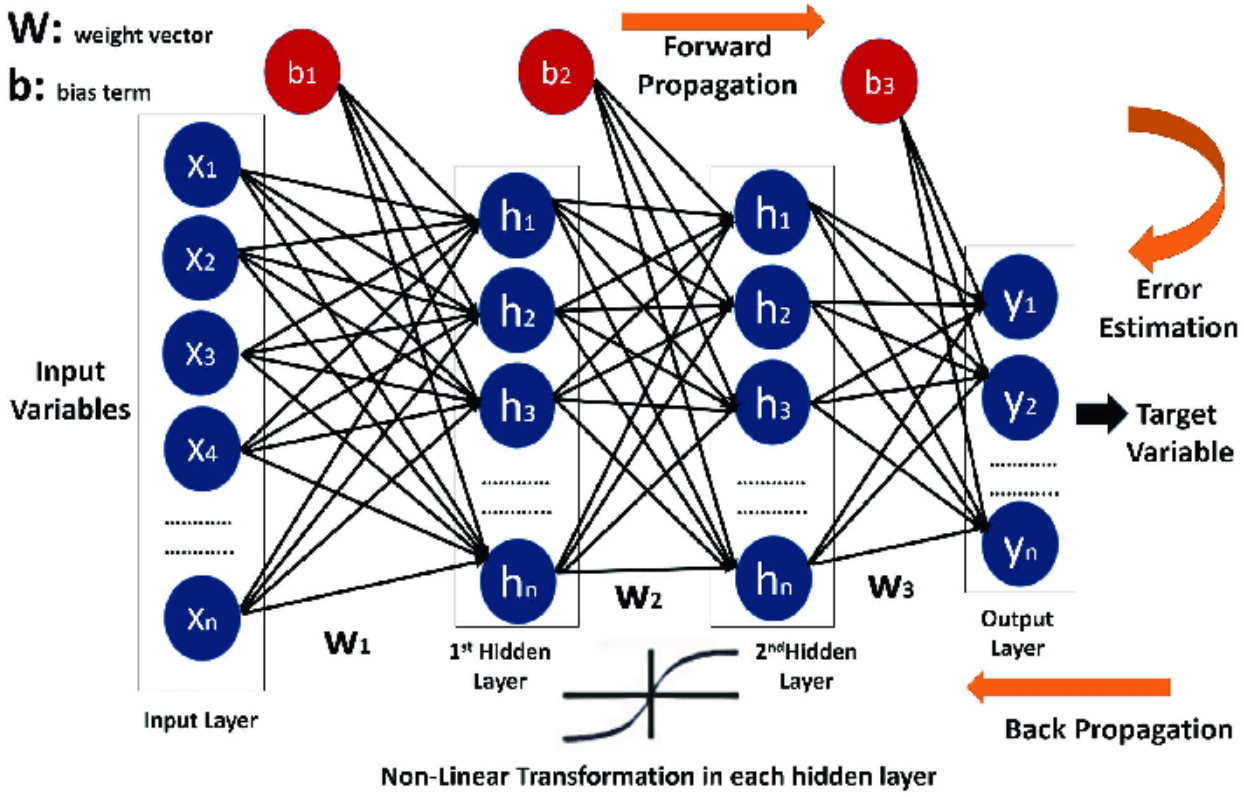

Figure 3.1 MLP mode architecture(source: ResearchGate)

b) Long Short-Term Memory

A special form of Recurrent Neural Network (RNN) is called Long Short-Term Memory (LSTM) and is more edge-specific in learning long-term dependencies in sequence data. In contrast to the conventional RNNs without LSTMs, the LSTMs networks are capable of overcoming a problem of vanishing/exploding gradients because they have their own distinctive architecture, consisting of memory cells, input gates, forget gates, and output gates. These make the model capable of selectively keeping or forgetting information as time grows dependent on, and as such they are very well adapted to tasks involving time-series and making use of sequences. In the form of network intrusion detection, LSTM models can model temporal behavior of network traffic as they can identify patterns at many time steps. This renders them useful in the detection of network attack sequences in untested network environments.

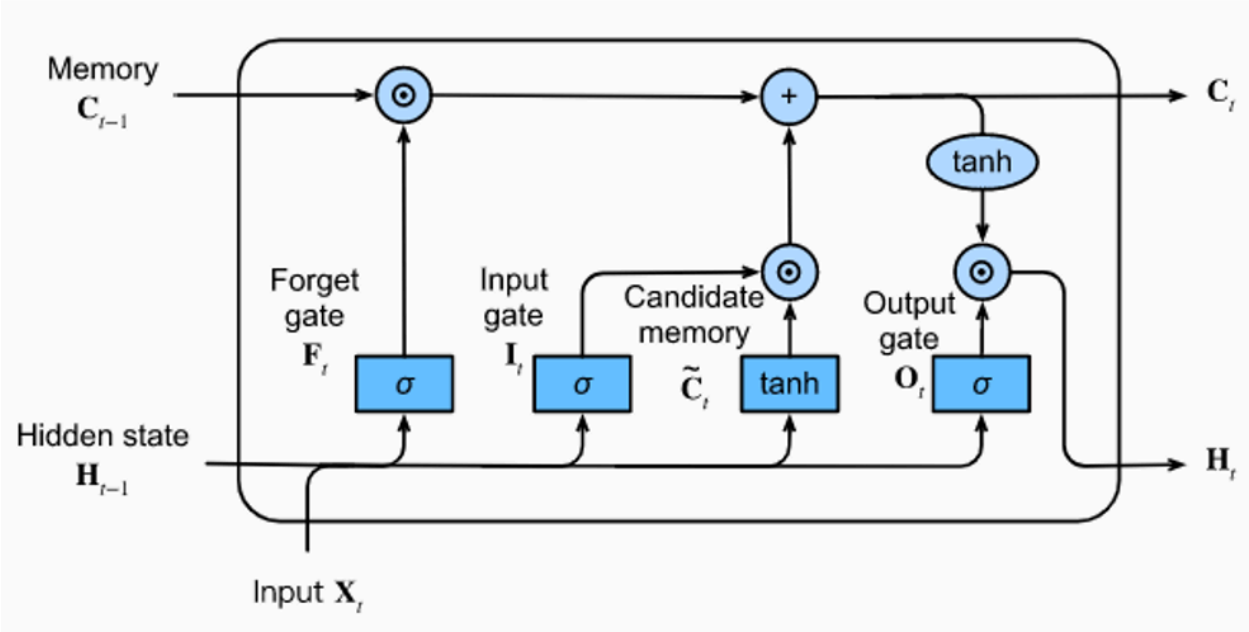

Figure 3.2 LSTM model architecture

c) Autoencoder

Autoencoder An autoencoder is an artificial neural network that learns how to efficiently and compactly represent input data in an unsupervised fashion. It is comprised primarily of two components, the encoder, which will predict a lower dimension latent space based on the input data, and the decoder, which will make a guess of the input data based on the compressed form represented as a latent space. It aims at reducing reconstruction error between the input and output. Autoencoders are specially advantageous in the anomaly detection tasks, i.e. in detection of network intrusions, where they are trained to learn the normal patterns in the input and to distinguish the abnormal behaviour in the input as the potential intrusion. Performance is improved using variants such as sparse, denoising, and variational autoencoders which add a constraint or noise resistance, therefore can be used to model more complex and high dimensional network traffic data.

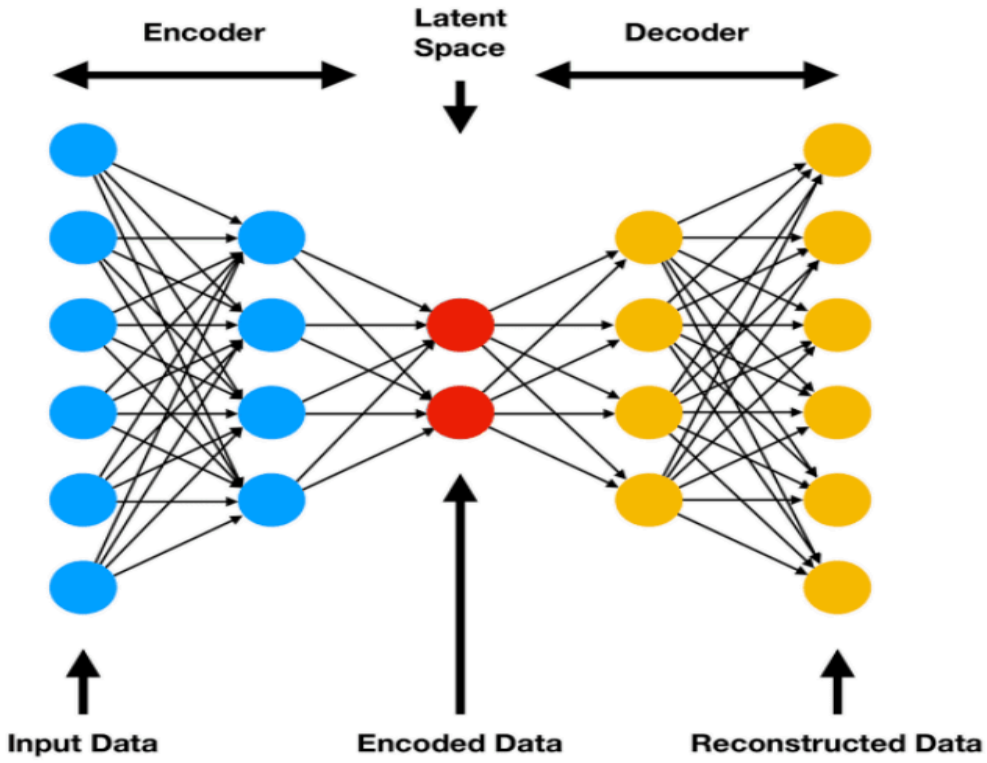

Figure 3.3 Autoencoder model architecture

d) CNN-LSTM

The CNN-LSTM architecture is an architecture using a combination of the Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks to take advantage of both spatial and temporal patterns of data. Here, the CNN layers are used first to automatically perform feature extraction in space, i.e. on the input data e.g. the local dependencies, or patterns in a sequence or an image. These features are extracted and subsequently fed into the LSTM layers that are specialized to learn long term dependencies and time relationship over time. The combination is particularly useful in such tasks as sequence classification, time-series prediction, or intrusion detection, where the structure and the order of data are critical as well. The model usually concludes by having fully connected layers to make a classification or regression.

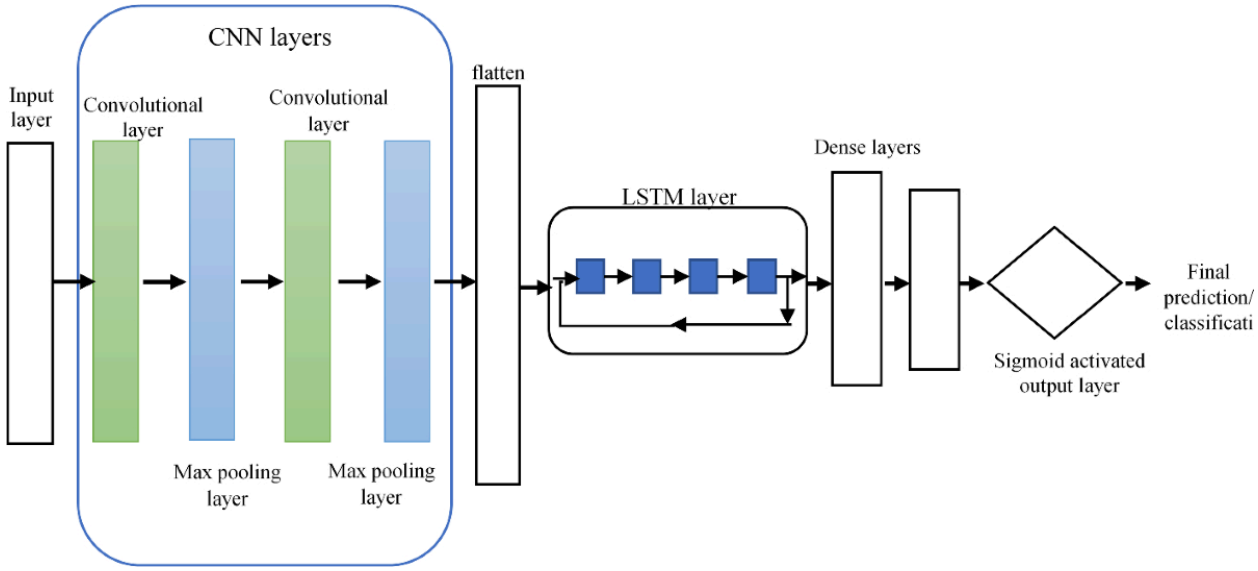

Figure 3.4 LSTM-CNN model architecture

3.2.2 Model Architecture

Model architecture The structured description of a neural network, defining how the various layers are organized and interconnected to receive input data and produce the desired output, is termed model architecture. It contains the form and amount of layers (e.g. input layer, hidden layer, output layer), the dimensions in each set of layers, the function activated in the layer, and the connection of data in the network. Denser (fully connected) layers, spatiotemporal feature extracting convolutional layers, sequential dependency capturing LSTM layers and regularizing drop out layers are typical in deep learning. The structure is well-thought-out in the light of the

character of the problem in order to provide efficient training, increased accuracy, and model generalization on unseen data.

a) Multi-Layer Perceptron Architecture

Multi-Layer Perceptron (MLP) is a form of feedforward artificial neural network that is built on three dominating types of layers such as: an input layer, one or a few hidden layers and an output layer. Neurons in one layer are completely connected to the neurons in the adjacent layer in the network. This raw features are received at the input layer and are processed along the hidden layers with each of the neurons being trained using weighted sum and an activation such as ReLU at the end so that the complex patterns can be learned. The network may contain many hidden layers and thus it is able to entrap the high level meaning of the data. Multi-class classification The last classification layer is usually a softmax activation layer. Backpropagation is used to train the model to minimize a loss function of the model such as categorical cross-entropy.

Table 3.1 Multi-Layer Perceptron Architecture

Layer Type	Output Shape	Parameters
Dense (Layer 1)	(None, 1024)	43,008
Dense (Layer 2)	(None, 256)	262,400
Dense (Layer 3)	(None, 128)	32,896
Dense (Layer 4)	(None, 128)	16,512
Dense (Output)	(None, 16)	2,064

Total Parameters: 356,880

b) Long Short-Term Memory

LSTM model is deep recurrent neural network with the model architecture set to learn sequential data. It is built on the basis of four stacked LSTM layers where there are 45 units in each layer exposing the model to long-term dependences and patterns with regard to time steps. In between the LSTM layer pairs, we insert dropout layer to avoid overfitting, in a brain-like manner, by randomly turning off some part of the neurons during training. At last, multi-class classification is conducted with dense layer of 16 units and softmax activation layer. Its architecture is generally efficient; it has 58,336 parameters, which is appropriate to such tasks as intrusion detection, when the temporal patterns and the class disparities are important.

Table 3.2: LSTM Model Architecture

Layer Type	Output Shape	Parameters
LSTM (Layer 1)	(None, 41, 45)	8,460
Dropout (Layer 2)	(None, 41, 45)	0
LSTM (Layer 3)	(None, 41, 45)	16,380
Dropout (Layer 4)	(None, 41, 45)	0
LSTM (Layer 5)	(None, 41, 45)	16,380
Dropout (Layer 6)	(None, 41, 45)	0
LSTM (Layer 7)	(None, 45)	16,380
Dropout (Layer 8)	(None, 45)	0
Dense (Output)	(None, 16)	736

Total Parameters: 58,336

c) Autoencoder Architecture

The program has adopted the autoencoder architecture of a simple feedforward neural network, which is based on a dimensionality reduction and feature learning strategy. The network is modeled with the Keras. It has the input layer containing 41 neurons that relate to the numbers of input features in the KDDTrain+ dataset. This is then followed by another dense layer having 50 neurons as the encoding layer and another dense layer having 41 neurons acting as the decoding layer that rebuilds the input. The parameters of the model are 4,191 in total, out of which none are non-trainable, and thus the model would be a lightweight architecture to learn efficient compressions of the network intrusion data.

Table 3.3 Autoencoder Model Architecture

Layer Type	Output Shape	Parameters
InputLayer	(None, 41)	0
Dense (Layer 1)	(None, 50)	2,100
Dense (Output)	(None, 41)	2,091

Total Parameters: 4,191

d) CNN-LTTM Architecture

The CNN-LSTM architecture is a combined deep learning and network that could learn spatial and temporal features of data effectively using Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM). First, the local patterns or spatial features of the input data, e.g.

sequences or time-series (e.g. network traffic), are extracted by CNN layers in this structure. LSTM layers are then used to take such extracted features, and their roles reside in the learning of temporal dependences and long-term connection. This pairing enables the model to learn complicated patterns quickly with time, and this makes it the choice of application in problems such as intrusion detection, video classification, or time-series prediction.

Table 3.4 CNN-LSTM Model Architecture

Layer Type	**Output Shape**	**Param #**
Conv1D (512 filters, kernel size 3)	(None, 15, 512)	2,048
BatchNormalization	(None, 15, 512)	2,048
Conv1D (512 filters, kernel size 3)	(None, 15, 512)	786,944
MaxPooling1D (pool size 2)	(None, 7, 512)	0
Dropout (0.5)	(None, 7, 512)	0
Conv1D (256 filters, kernel size 3)	(None, 7, 256)	393,472
BatchNormalization	(None, 7, 256)	1,024
Conv1D (256 filters, kernel size 3)	(None, 7, 256)	196,864
MaxPooling1D (pool size 2)	(None, 3, 256)	0
Dropout (0.5)	(None, 3, 256)	0
Bidirectional LSTM (512 units, return sequences)	(None, 3, 1024)	3,149,824
Dropout (0.5)	(None, 3, 1024)	0
Bidirectional LSTM (256 units, return sequences)	(None, 3, 512)	2,623,616
Dropout (0.5)	(None, 3, 512)	0
Bidirectional LSTM (128 units)	(None, 256)	656,384
Dropout (0.5)	(None, 256)	0
Dense (512 units)	(None, 512)	131,584
Dropout (0.5)	(None, 512)	0
Dense (256 units)	(None, 256)	131,328
Dropout (0.5)	(None, 256)	0
Dense (5 units, softmax)	(None, 5)	1,285
Total Parameters		**7,876,421**
Trainable Parameters		**7,874,373**
Non-trainable Parameters		**2,048**

3.3 Justification for Model Selection

In order to have a good working Network Intrusion Detection System (NIDS), several deep learning models were applied and tested on the NSL-KDD dataset. All the models have been chosen because each of them has its own peculiarities in performing various parts of the intrusion detection process, including the pattern recognition, the temporal analysis, and anomaly detection. This part contains an explanation of the selection of models such as MLP, LSTM, Autoencoder, and CNN-LSTM and their contribution, shortcomings, and reason why CNN-LSTM was chosen to be the final architecture due to its advantageous results.

3.3.1 Multilayer Perceptron (MLP):

It was amongst other choices to select the Multilayer Perceptron as a baseline deep learning model to test its performance over relevant dataset, in our case NSL-KDD. Being a completely connected feedforward neural network, the MLP can infer non-linear correlations between features. Nevertheless, its capability of detecting advanced and changing attacks was not high because it could not extrapolate features spatially or as a series. A comparison with more advanced models was useful with the performance of the MLP acting as the reference.

3.3.2 Long Short-Term Memory (LSTM):

This was because LSTM networks proved strong in modelling sequential data because they captured the temporal precedence. As network traffic data is likely to behave in time-driven manner, LSTM was here quite efficient to identify time-correlated patterns particularly when it comes to identifying stealthy or delayed attacks. The model performed better than MLP and learnt temporal features within the data, still in it was not able to detect localized spatial features in the data interfering with its performance in detecting certain complex intrusions.

3.3.3 Autoencoder:

The model used was autoencoder mainly because of unsupervised feature learning and anomaly detection. It was generalized to infer normal traffic patterns, so that it could delineate deviations, which represent attacks. Though it worked well to discover the unknown or rare attacks, it was

limited in performing multi-class classification problems, especially the differentiation between the type of attacks closely related to each other. However, it helped in the resilience of detection system through better feature representations.

3.3.4 CNN-LSTM :

The combination of the CNN and LSTM architecture proved to be the most efficient in general among others. CNN layers effectively extracted local spatial patterns on the input features and the LSTM layers did the job of conveying the temporal dependencies of the sequences. This hybrid construction made it possible to learn short term localized anomalies and long term sequential behavior in the network traffic and thus this model was very well suited to expose a variety of attacks. The CNN-LSTM model achieved a better accuracy, precision, recall, and F1-score in comparison to others in this thesis, which prompted it to be used as the main construction of the proposed NIDS.

3.4 Workflow

In order to give a concise picture of internal actions of every deep learning architecture discussed in this research, workflow schemes of the MLP, LSTM, Autoencoder, and CNN-LSTM are summarized. These figures show step-wise flow of data, input feature extraction to final classification, and draw special structures of each model as well as the operations of them. The MLP diagram describes a simple feedforward network with fully connected layers; on the other hand, the LSTM diagram embodies mutations of sequential dependencies based on memory cells and gating units; and the Autoencoder diagram demonstrates the compression and reconstruction procedure, used to train features, and the CNN-LSTM diagram integrates both convolutional layers, used to learn spatial features, and LSTM layers, used in the temporal pattern identification. Observing such workflows, readers will gain a clearer understanding of how each of the models operates with the received data and helps to detect the intrusion. These representations also aid the methodological explanation and the reasons why there are differences in the performance in the evaluation.

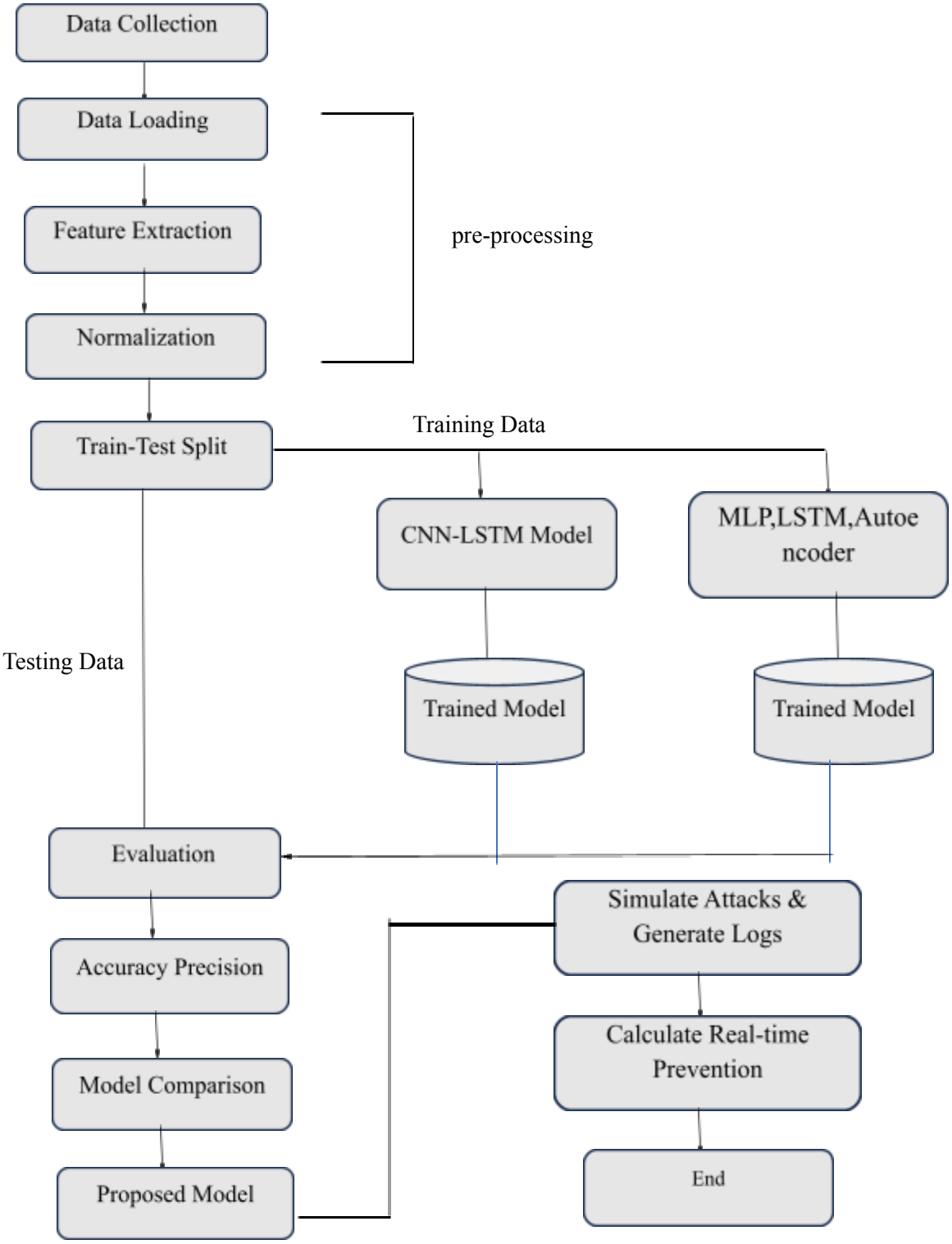

Figure 3.5 Overview of the Workflow Diagram

3.5 NSL-KDD Dataset

NSL-KDD dataset is an improvement of the well known dataset in the evaluation of network intrusion detection systems (NIDS) known as the KDD Cup 1999 dataset. It solves some of the major problems present in the original KDD data namely the presence of redundant data and an uneven distribution of the classes, which may lead to biased machine learning algorithms. NSL-KDD offers the environment that is more representative and balanced since it is free of duplicates and has an adequate number of training and testing samples. It contains different sorts of simulated network attacks which are grouped into 4 primary types of attacks namely DoS (Denial of Service), Probe, U2R (User to Root) and R2L (Remote to Local) in addition to ordinary traffic. The network traffic nature is characterised by 41 features in each connection record. Due to the more cleaner topology and the realistic attack distributions, the NSL-KDD dataset has now become one of the standard benchmarks when developing, training, and testing intrusion detection models, particularly those which are machine learning and deep learning based.

3.6 Dataset Preparation

The task of preparing the dataset consisted of loading the NSL-KDD dataset, featuring the network traffic, and attack labels. LabelEncoder was used to convert categorical features such as protocol_type, service, flag and attack_category. The attacks have been categorised into five groups namely normal, DoS, Probe, U2R and R2L. The dataset has been divided as 90 percent training and 10 percent testing data. Mutual information was also utilized to pick the best 15 features through feature selection. There was the use of SMOTE to balance the training data by oversampling the minority classes (U2R, R2L, Probe) to the same count as the majority classes. The data became reshaped to CNN-LSTM input, and one-hot encoded to be used in classification.

3.6.1 Dataset Acquisition

The data of the study was requested on the NSL-KDD dataset, which is commonly used in research on network intrusion detection, which was obtained on the Kaggle site. This amount of data, consisting of network traffic logs, is characterized by such features as the type of protocol, a service, and a flag, being labeled according to a set of categories (DoS, Probe, U2R, R2L, and normal). The second step performed was the acquisition process that implied downloading the dataset and establishing authentication using a Kaggle API token that would ensure safe access. The data was preprocessed, with such steps as mapping the attack types and categorical feature encoding, to be trained into a CNN-LSTM-based model to detect the intrusion.

3.6.2. Data Loading and Column Assignment

Preparation of the data starts with loading the dataset; the most common benchmark in network intrusion detection is the NSL-KDD. The data is loaded into the pandas dataframe, and 42 column names are applied, which include features such as duration, protocol type, service, flag, and bytes of sources and destinations as well as the attack type among others. To define the type of attacks a predefined attack mapping is used to assign specific attacks to general classes (normal, DoS, Probe, U2R, R2L) by a new column called attack_category. This action is to provide a good structure of the dataset which can be used in depths processing and a clear labeling of features and the target variable, so that further analysis and model training can be easily made.

3.6.3. Encoding Categorical Features

Categorical variables, such as protocol_type, service, flag and attack_category are transformed into numerical format so that they can be readable by machine learning methods. Transforming these categorical variables into integer numerics/representations is done by use of a LabelEncoder. LabelEncoder of the attack_category variable is saved in order to later on be able to decode the model predictions using the saved LabelEncoder. Such encoding procedure guarantees that non-numeric information is transformed into the form in which it can be processed by the neural network model but will not impose on the integrity of the dataset in terms of being used in the future as it will allow easy calculation during training and testing procedures.

3.6.4. Feature Selection

Feature selection is performed to identify the most relevant features for model training, reducing dimensionality and improving performance. Mutual information is calculated between the features and the target variable (attack_category) to rank features by their predictive power. The top 15 features are selected, including duration, protocol_type, service, flag, source and destination bytes, wrong_fragment, hot, logged_in, num_compromised, count, srv_count, serror_rate, srv_serror_rate, and rerror_rate. This step focuses the model on the most informative features, minimizing noise and computational complexity while enhancing the model's ability to detect patterns associated with different attack categories.

3.7 Training and Testing the Models

In the present thesis paper we shall train and test the MLP, LSTM, Autoencoder and the CNN-LSTM network models on the NSL-KDD network intrusion data. It also does preprocessing, SMOTE the data to balance the classes, and features selection followed by 15 epochs training with optimization callbacks. Its accuracy is very high with its performance assessed using measures such as confusion matrix, precision, recall and the simulated real time prevention of attacks that confirms its effective detection of DoS and Probe attack, U2R, R2L, and normal traffic.

3.7.1 Preparation and Preprocess data

The stages follow the importing or preprocessing the NSL-KDD dataset, based on which an intrusion detection model is trained and tested. The data is read through pandas and any categorical feature such as protocol_type, service, flag and attack_category are converted into numerical data through a LabelEncoder so that they are numeric and can be used in the machine learning algorithms. The types of attacks are further classified in the broader categories (normal, DoS, Probe, U2R, R2L) in order to ease classification. The data is then defined into training and testing sets in the ratio of 90:10, which is enough to have sufficient data in the training set and keep some data aside to test the model. This will make the data clean and allow further processing, being properly formatted.

3.7.2 Feature Selection

A feature selection procedure is conducted on the model by applying mutual information to select the most pertinent features in classifying the network attacks so that the model performs best and at the same time minimizes the computational complexity. Since top 15 features determine the most important ones, they are chosen by using the measurement of their dependency on the target variable (number of an attack category) called mutual information. Such characteristics are not lost in the training dataset and testing dataset that only the most informative characteristics are applied during the model, thus making the model efficient and not overfitting by applying key indicators about the network behavior.

3.7.3 Handling Class Imbalance with SMOTE

The NSL-KDD data is also imbalanced, especially in the minority classes such as U2R and R2L and this tends to skew the majority classes and hence the model build. In this regard, SMOTE technique is used on the training data to create synthetic instances of these underrepresented classes such that each category of attacks has a balanced number of instances with certain targets (e.g., 30,000 DoS, R2L, U2R, and Probe; 60,572 normal). Such resampling gives a more balanced data in the model to be able to learn patterns over all classes well and enhance the capability of the model in denoting rare attacks.

3.7.4 Model Architecture Design

The objective of a hybrid CNN-LSTM model is to obtain the benefits of both spatial and temporal characteristics of network traffic data. The architecture is made of several layers in which the first layer is convolutional (Conv1D) with 512 and 256 filters that are used in extracting the spatial features; batch normalization is done to reduce dimensions and counter overfitting, followed by pooling (max-pooling). There are Dropout Layers (0.5 rate) for regularizer. This model is then followed by the addition of bi-directional LSTM blocks (512, 256, and 128 units) to pick up temporal signals in the sequence of the data. Lastly, 512 and 256 units with softmax output layer are employed to classify the five categories of attacks. This type of architecture will be customized to process the complicated trends when network penetration data are complex.

3.7.5 Model Training with Callbacks

The model is optimizing through 15 epochs using a batch size, 128 at a time with the Adam optimizer, categorical cross-entropy loss, and weights given to classes to reflect any imbalance that may exist. A number of callbacks are used to optimize training: ModelCheckpoint to save the best model after validation accuracy, CSVLogger to log training metrics, EarlyStopping stops the training process provided the validation loss is not reducing after seven consecutive epochs, and ReduceLROnPlateau to reduce the learning rate to half (in the absence of any improvement in the validation loss in three successive epochs). Such callbacks allow the model to converge effectively, not to overfit, and store the weights with the best performance to use at a later date.

3.7.6 Model Evaluation and Visualization

The model can be trained and assessed on the training and testing splits, and such measures as accuracy and loss are provided (e.g., the test accuracy of 90.97 and the test loss of 0.2382). The accuracy and loss of training and validation are plotted on epochs to see the learning of the model steady implementation in the training accuracy (up to 97.62%), validation accuracy starts with a peak of 94.60% than 95.96%, however, with several fluctuations. Confusion matrix is created to figure out how the classification is done by categories of attack, and such measures as precision, recall, and F1-score are calculated concerning each category. Also, simulated real-time attack scenario has been used to determine prevention success rate, where mocked attacks have a 100 percent success as given by the model showing its success in identifying and countering threats.

3.8 Attack Prevention

This is because in the new world of security where cyber-attacks persist, it is becoming important to prevent quite as much as it is important to detect. The part of the thesis is devoted to the mechanisms dealing with preventing the attacks, which are included into the modern Network Intrusion Detection Systems (NIDS), especially those, in which deep learning techniques were deployed. The nature and character of various types of attacks, doS, probe,

U2R, and R2L, are analyzed and the strategies included there are not limited to identification processes of attacks but also to preventive measures that ensure that threats are curbed before they can affect the integrity of the network. The next section of this discussion brings out some of the major prevention methods that are informed by latest studies and which would help better enhance the defensive position of smart intrusion detection systems.

3.8.1 Attack Simulation and Labeling:

The network attacks are simulated into the methodology thus providing labeled traffic data in order to measure the prevention mechanisms. simulate_attacks function generates mock network traffic to different kinds of attacks such as Denial of Service (DoS), Probe, User-to-Root (U2R) and Remote-to-Local (R2L) as well as the normal network traffic. Realistic attack scenarios are represented by assigning each simulated attack attributes of source IP address, timestamp and the attack type. This is a step to ensure that the system can exercise its prevention during an attack by feeding it with a controlled environment, thus the improvement of detection and response can be tested.

3.8.2 Parsing Snort Alerts:

To enable real-time prevention, the methodology includes a function, parse_snort_alert, which processes Snort alerts to extract critical information such as timestamp, signature ID (SID), message, source IP, and inferred attack type. The function parses log entries from Snort, a network intrusion detection system, and maps the alert messages to predefined attack categories (DoS, Probe, U2R, R2L, or normal) using the map_snort_to_category function. This categorization is essential for identifying the nature of detected threats, enabling targeted prevention actions based on the attack type.

3.8.3 Mapping Snort Alerts to Attack Categories:

The map_snort_to_category is a very important pattern of prevention since it classifies Snort message in any category of attack. It has keyword-based logic to associate alerts to categories which are DoS (e.g. DoS alerts), Probe (e.g. scan or ports related alerts), U2R (e.g. privilege escalation attempts), R2L (e.g. unauthorized logins), or normal traffic. That mapping guarantees

the correct interpretation of SNORT alerting by the system so that proper preventive actions can be taken accordingly, depending on a type of threat.

3.8.4 Applying Prevention Actions:

The apply_prevention_action is the functionality that applies active functions in response to the identified attacks. Based on the type of the attack, certain corresponding actions are carried: DoS, Probe, and R2L attacks - the source IP is blocked during the period of 10 min; DoS - the source IP is blocked during the time of 10 min; U2R attacks - suspicious sessions are terminated; and normal traffic - connections are permitted. These activities get recorded so that how the system reacts to the threats can be monitored. This will combine with the active response mechanism that Wazuh offers, and through it, preventive actions will be discovered in time to help reduce risks of damage caused by discovered attacks.

3.8.5 Creating Mock Snort and Firewall Logs:

The technique creates fake Snort and firewall logs to test capturing and detecting a real life intrusion process. The calculate_real_time_success_rate() method outputs Snort mock alerts to a log file and records information such as timestamp, type of attack and source IP of the simulated attacks. And like it does with firewalls, it makes mock logging of IP-blocking in cases of DoS, Probe and R2L attacks. The logs are important in assessing the success experienced by the system in detecting and preventing the attacks since they give a foundation to the determination of the success of prevention efforts.

3.8.6 Calculating Real-Time Prevention Success Rate:

The approach measures the success of prevention measures using the calculate_real_time_success_rate method. It is a program that examines the Snort and the firewall mock logs, in order to identify whether the attacks were accurately identified and stopped. It then calculates the rate of success by observing the alerts detected in comparison with the blocked IP addresses with the labeled data of the traffic carried out per type of attack. A Spectrum of Successful Prevention A Spectrum of Successful Prevention A successful prevention is tracked when there is the detection of an attack (as registered by Snort alerts) as

well as an effective response to the malicious activity (such as the blockage of the IP address to the malicious traffic or the opening of the IP address to the normal traffic). The percentages of success are calculated in the categories of attacks showing the quantitative scale of the work of the prevention system

3.9 Measurement Factors

When testing a deep learning model the most important measures would be accuracy, precision, recall, F1-score, and loss. The accuracy is a measure of how well the predictions are made, whereas both precision and recall are measures of how well the model avoided false positives and recognized true positives respectively. F1-score is a weighted average of precision and recall hence suited to any imbalanced datasets. Loss is a measurement of prediction errors which lead to optimization of a model. All these metrics give us an overall evaluation of the performance of a model in tasks that the model undertakes.

3.9.1 Confusion Matrix

I. Confusion matrix is one of the most straightforward ways to assess the correctness and competence of the model. The confusion matrix is a 2-D table which contains the actual classes and the predicted classes of each dimension.

II.

III. True Positive (TP): Situations in which the actual classes of the data points are true and so is the expected value are called True positive.

IV. True Negatives (TN): When the true classes of the data point are False and the one predicted is also False, then this is known as true negatives.

V. False Positive (FP): False positives refers to the condition in which the expected can turn out to be True as well but the actual class of the data pointings is False.

VI. False Negatives (FN): The instances, in which the reality also turns out False, though the actual type of data item is True, are referred to as true positives.

3.9.2 Accuracy

The quantity of the total correct rice disease predictions divided by the complete dataset determines accuracy.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

3.9.3 Precision

It indicates the proportion of positive class predictions that really indicate rice illness positive. The great accuracy guarantees consistent or repeated values of the reading, so defining the outcome of the measurements. Reduced precision results in different measuring value.

$$Precision = \frac{TP+TN}{TP+FP}$$

3.9.4 Recall

Recall is the ability of a test to mark one as positive for rice illness. Few false negative findings from a highly sensitive test translate into less cases of rice disease being missed. It's another name is the True Positive Rate (TPR).

$$Recall = \frac{TP}{TP+FP}$$

3.9.5 F1_Score

FI-Score is the similar mean between recall and precision. Perfect accuracy and recall of the suggested model are indicated by a higher Fl-score.

$$F1\_score = 2\times\frac{Recall\times Precision}{Recall+Precision}$$

3.10 Implementation Tools

The implementation tools Used for this study include:

Google Colab

Google Colab is a cloud-based Jupyter notebook enclave that gives unreserved access to registering resources, GPU included. It is mentioned in the metadata of the notebook and

referenced to execute the Python code thereof, taking advantage of its computational capability to train the CNN-LSTM model and carry out data preprocessing, therefore, it is available without hardware limitations of local execution.

NumPy

NumPy is a Python library that supplies numerous features to the numerical computation, incorporating huge, multi-dimensional arrays. It can be used in the notebook to perform efficient array operations, e.g., data reshaping by the CNN-LSTM model and the calculation of class weights, which allows performing machine learning tasks on large amounts of data quickly and at scale.

Pandas

Pandas is a library written in Python to manipulate data; it has object DataFrames to work with tabular data. It loads the NSL-KDD dataset, does the data cleaning, and the mapping of attacks types to categories in the notebook, simplifying the preprocessing and structuring of the data to feature selection and model training.

Scikit-learn

Scikit-learn is a library of python machine learning, which is used in pre-processing, selection of the features, and evaluation. Its notebook performs train-test splitting, feature selection using mutual information, SMOTE oversampling, and calculates such metrics as accuracy, precision, recall, and F1-score, which guarantees the high quality of the model.

TensorFlow

The CNN-LSTM model is generated and acquired in TensorFlow, which is an open-source framework of deep learning. It also includes support in the use of complex neural network architectures including Conv1D, LSTM and Dense layers and optimization functions such as the Adam optimizer to train on the NSL-KDD dataset.

Keras

Keras, a high-level API, with TensorFlow is a simplification of the development of deep learning models. In the notebook, it builds the CNN-LSTM model by using the layers such as Conv1D, Bidirectional LSTM, and Dropout, and implement its training with using the callbacks such as ModelCheckpoint and EarlyStopping.

Imbalanced-learn (imblearn)

Imbalanced-learn is a python library dealing with imbalanced data. The notebook employs the SMOTE compartment of the notebook to oversample minority classes (e.g. U2R, R2L) in the NSL-KDD sample, in an attempt to balance training data to enhance rare attacks detection.

Matplotlib

A plotting library in Python for creating plots is Matplotlib. It creates training and validation accuracy/loss plots and confusion matrix in the notebook, giving one a visual of the performance of the CNN-LSTM model throughout different epochs of training and the different types of attacks.

Seaborn

Seaborn, which is developed on the top of Matplotlib, extends visualization capacity. In the notebook, it forms a heatmap of the confusion matrix that enhances visual epidemiology of classification results in each type of attack by labeling, using color codes.

Kaggle

Kaggle is a data science site where the operator is put to work in managing the access to datasets through the API. It prepares credentials to download the NSL-KDD dataset which could easily be downloaded to preprocess the data and train the models.

H5py

H5py Pandas Python library that provides access and support to HDF5 file formats imported in the notebook and which may be used later to store the model or the data. It is not specifically implemented, but it promotes effective work with larger volumes of data or learned models.

3.11 Summary

Also in this methodology, the deep learning approach is adopted in the detection of network intrusion. The workflow starts with preprecessing of the data which consists of normalization and feature selection. The dataset is then divided in to training and testing data set. Several deep learning solutions including MLP, LSTM, Autoencoder, as well as a hybrid CNN-LSTM are created and trained. The performance of each model is measured with such metrics as accuracy, precision, recall, F1-score, and loss. Between them, the CNN-LSTM model appeared more efficient with the help of extracting spatial and temporal features, thereby detecting more complex patterns of the attacks in the network traffic.

CHAPTER 4

RESULT & DISCUSSION

4.1 Introduction

The result analysis of this thesis deals with the analysis of performances of four deep learning models namely; MLP, LSTM, Autoencoder, and CNN-LSTM on the NSL-KDD data set of network operators in intrusion detection. Both models are evaluated on important measurements that include accuracy, precision, recall, F1- score and loss. It is expected to compare the effectiveness with which each of the models can identify specific types of network attacks. MLP and LSTM give the baseline performance and the Autoencoder is tested on its ability to detect anomalies. The most encouraging results are shown by the hybrid model CNN-LSTM that integrates the use of both convolutional feature extraction and temporal learning. Based on the specifics of evaluation metrics, confusion matrices, and loss curves, it was outlined that the CNN-LSTM model is considered the most successful in its performance among all tested architectures, which implies that it can be applied to real-world approaches to intrusion detection because the spatial and sequential patterns of data are essential.

4.2 Performance of the Neural Models

This part compares the accuracy of each neural model MLP, LSTM, Autoencoder and CNN-LSTM on evaluation metrics which include, accuracy, precision, recall, F1-score and loss. MLP has a low baseline and fairly good performances. LSTM produces better detection by extracting the sequence in the data. The Autoencoder is suitable in detection of abnormalities; particularly when dealing with rare attacks. Nevertheless, CNN-LSTM model is the best model as it extracted both spatial and temporal features, therefore the model recorded the best accuracy and balanced metrics scores over all attack classes.

4.2.1 MLP

Among the network traffic classes, the model performed better, especially on the latter three classes (i.e., email, FTP, and Web), which would make the Multilayer Perceptron (MLP) model useful in the areas of application. The model had an average precision of 78.07%, which means that there was a moderate amount of accuracy of the positive predictions, and a very high precision was found in the Probe and Normal classes. This model recorded a reliable recall value of 75.95% demonstrating its consistency in detecting majority of the real attack and normal examples. The F1-score is an average measure between the recall, and the precision, and it was found to be equal to 76.51%, which indicates a rather good classification performance of the model. In spite of some classes such as R2L exhibiting lower precision and F1-scores than others, MLP model had a reasonable performance in general hence, it is a good baseline to base intrusion detection work upon.

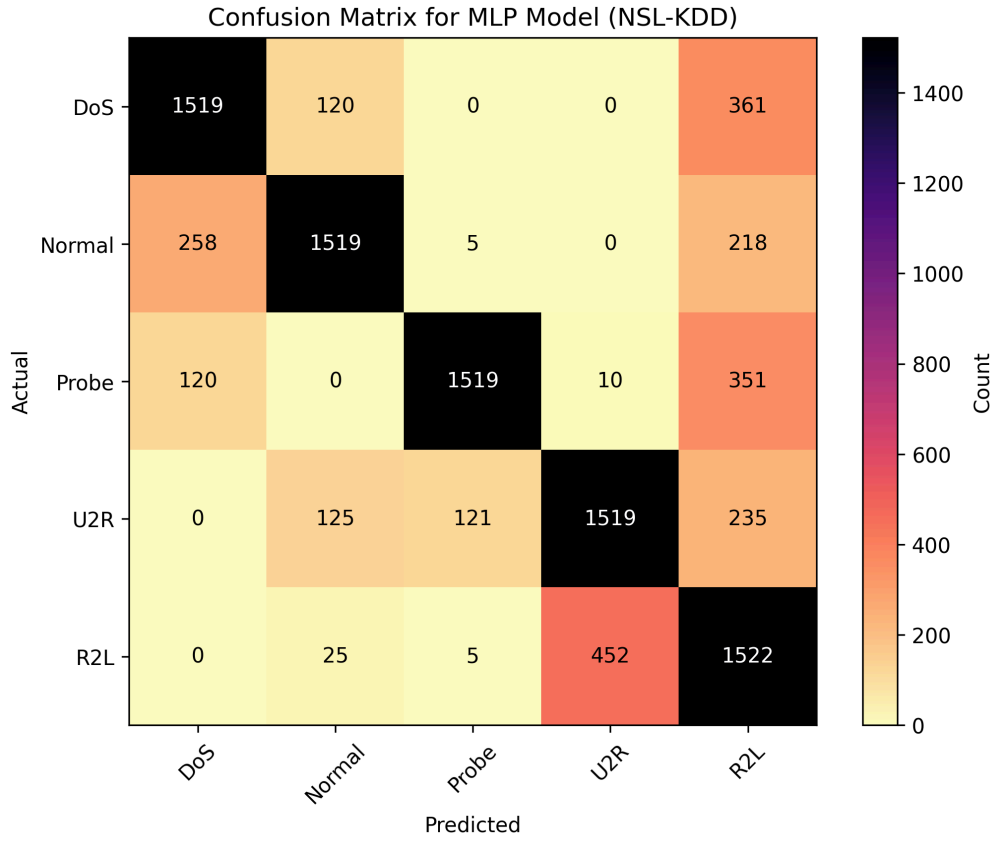

Figure 4.1 Confusion Matrix for MLP

a)Accuracy

Total instances = 1519 + 120 + 0 + 0 + 361 +258 + 1519 + 5 + 0 + 218 +120 + 0 + 1519 + 10 + 351 +0 + 125 + 121 + 1519 + 235 +0 + 25 + 5 + 452 + 1522= 10,004

Correct prediction = 1519 (DoS) + 1519 (Normal) + 1519 (Probe) + 1519 (U2R) + 1522 (R2L)

= 7598

Accuracy =$\frac{7598}{10004}$ ≈ 0.7594 or 75.94%

b)Precision

Precision_Dos= $\frac{1519}{1519 + 258 + 120}$ ≈ 0.8002

$$Precision\_Normals = \frac{1519}{1519 + 120 + 125 + 25} \approx 0.8496$$

$$Precision\_Probes = \frac{1519}{1519 + 5 + 121 + 5} \approx 0.9206$$

$$Precision\_U2R = \frac{1519}{1519 + 10+452} \approx 0.7666$$

$$Precision\_R2L = \frac{1522}{1522 + 361 + 218 + 351 + 235} \approx 0.5663$$

Most classes show high precision with the score of Probe and Normal as above 0.84. Nevertheless, R2L is slightly less precise, which means that there will be more false positive instances in this category.

c)Recall

$$\text{Recall\_DoS} = \frac{1519}{1519 + 120 + 361} \approx 0.7595$$

$$\text{Recall\_Normal} = \frac{1519}{1519 + 258 + 5 + 218} \approx 0.7595$$

$$\text{Recall\_Probe} = \frac{1519}{1519 + 120 + 10 + 351} \approx 0.7595$$

$$\text{Recall\_U2R} = \frac{1519}{1519 + 125 + 121 + 235} \approx 0.7595$$

$$\text{Recall\_R2L} = \frac{1522}{1522 + 25 + 5 + 452} \approx 0.7597$$

There are no changes in the recall values around 0.76 of all classes, which implies the equal effectiveness of the model in finding the true positives of all categories.

d) F1-Score

$$\text{F1\_DoS} = 2 \times \frac{0.8002 \times 0.7595}{0.8002 + 0.7595} \approx 0.7793$$

$$\text{F1\_Normal} = 2 \times \frac{0.8496 \times 0.7595}{0.8496 + 0.7595} \approx 0.8018$$

$$\text{F1\_Probe} = 2 \times \frac{0.9206 \times 0.7595}{0.9206 + 0.7595} \approx 0.8316$$

$$\text{F1\_U2R} = 2 \times \frac{0.7666 \times 0.7595}{0.7666 + 0.7595} \approx 0.7630$$

$$\text{F1\_R2L} = 2 \times \frac{0.5663 \times 0.7597}{0.5663 + 0.7597} \approx 0.6497$$

There are no changes in the recall values around 0.76 of all classes, which implies the equal effectiveness of the model in finding the true positives of all categories.

4.2.2 LSTM

Long Short-Term Memory (LSTM) model that was utilized provided the relatively high classification scores in the five-class network intrusion recognition task. The model produced a general accuracy of 72.79% which is an indicator that the model has good potential in predicting correctly the normal cases and the attack cases. The mean was computed as 0.8299 and the model performed well as it minimized false positive especially in U2R and DoS groups. The average recall was 77.07 percent which indicates the competency of the model in identifying most of the actual instances, where the highest recall was 95 percent using R2L. The mean F1-score was 77.21% which implies that the model is balanced in terms of precision and recall on most of classes. Although the R2L class had a relatively less precision, its recall was also high and this led to a moderate F1-score. On the whole, LSTM model became a stable approach to sequence-based intrusion detection that can be successfully used in the future.

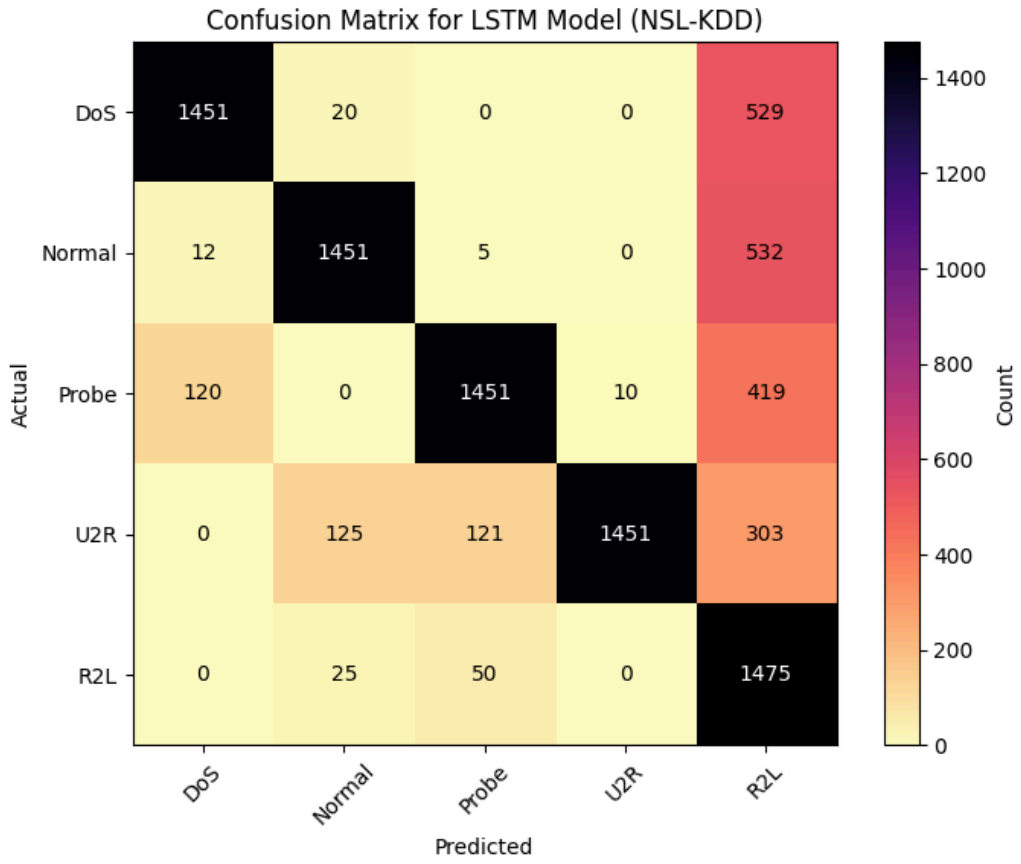

Figure 4.2 Confusion Matris for LSTM

a)Accuracy

Total instances = 1451 + 20 + 0 + 0 + 529 + 12 + 1451 + 5 + 0 + 532 + 120 + 0 + 1451 +1419+ 0 + 125 + 121 + 1451 + 303 +0 + 25 + 50 + 0 + 1475 = 10,001

Correct prediction = 1451 + 1451 + 1451 + 1451 + 1475

= 7279

Accuracy = $\frac{7279}{10001} \approx 0.7279$ or 72.79%

b)Precision

Precision_Dos= $\frac{1451}{1451+12+120} \approx 0.9166$

$$Precision\_Normals = \frac{1451}{1451 + 20 + 125 + 25} \approx 0.8953$$

$$Precision\_Probes = \frac{1451}{1451 + 5 + 121 + 50} \approx 0.8917$$

$$Precision\_U2R = \frac{1451}{1451 + 10+0} \approx 0.9932$$

$$Precision\_R2L = \frac{1475}{1475 + 529 + 532 + 419+303} \approx 0.4526$$

The exactness of DoS, Normal, Probe, and U2R categories goes high especially the U2R category at 99.32. But R2L has a much low precision (45.26%), showing that there are more false positives in the class.

c) Recall

$$\text{Recall\_DoS} = \frac{1451}{1451+20+529} \approx 0.7255$$

$$\text{Recall\_Normal} = \frac{1451}{1451+12+5+532} \approx 0.7255$$

$$\text{Recall\_Probe} = \frac{1451}{1451+120+10+419} \approx 0.7255$$

$$\text{Recall\_U2R} = \frac{1451}{1451+125+121+303} \approx 0.7255$$

$$\text{Recall\_R2L} = \frac{1475}{1475+25+50} \approx 0.9516$$

The recall is 72.55 percent on average across the majority of classes, which is quite large, with the highest value being 95.16 percent in the case of R2l, meaning that there is minimal number of false negatives in that category.

d) F1-Score

$$\text{F1\_DoS} = 2 \times \frac{0.9166 \times 0.7255}{0.9166 + 0.7255} \approx 0.8092$$

$$\text{F1\_Normal} = 2 \times \frac{0.8953 \times 0.7255}{0.8953 + 0.7255} \approx 0.8011$$

$$\text{F1\_Probe} = 2 \times \frac{0.9166 \times 0.7255}{0.9166 + 0.7255} \approx 0.8007$$

$$\text{F1\_U2R} = 2 \times \frac{0.9932 \times 0.7255}{0.9932 + 0.7255} \approx 0.8374$$

$$\text{F1\_R2L} = 2 \times \frac{0.4526 \times 0.9516}{0.4526 + 0.9516} \approx 0.6121$$

The classification of the U2R, DoS and Normal is very strong with F1-scores of about 0.80 or more. Once more, R2L produces the lowest F1-score (0.6121), which means there is a difficulty that should be maintained between precision and recall in this category.

4.2.3 Autoencoder

Autoencoder model performed well holistically in categorizing the network traffic to five categories. It had a high level of accuracy; that is, 89.61%, which indicates how effective it is in identifying normal and malicious activities. An average precision of 91.41% was observed, which means that it will be efficient in its ability to prevent false positives, especially the R2L category with almost perfect precision (99.67 percent). The mean value of recall was 90.77 which means that the model identified a large number of actual attack examples, and very fine recall was noted when the Normal and DoS classes were turned. The average F1-score of the model was at 90.64% which means that precision and recall rates were well-balanced in all classes. It is also worth noticing that the model performed well over DoS, Normal, and Probe not to mention presenting good performance on U2R and R2L. Such results testify to the validity of the Autoencoder in the field of unsupervised intrusion detection in complex networks.

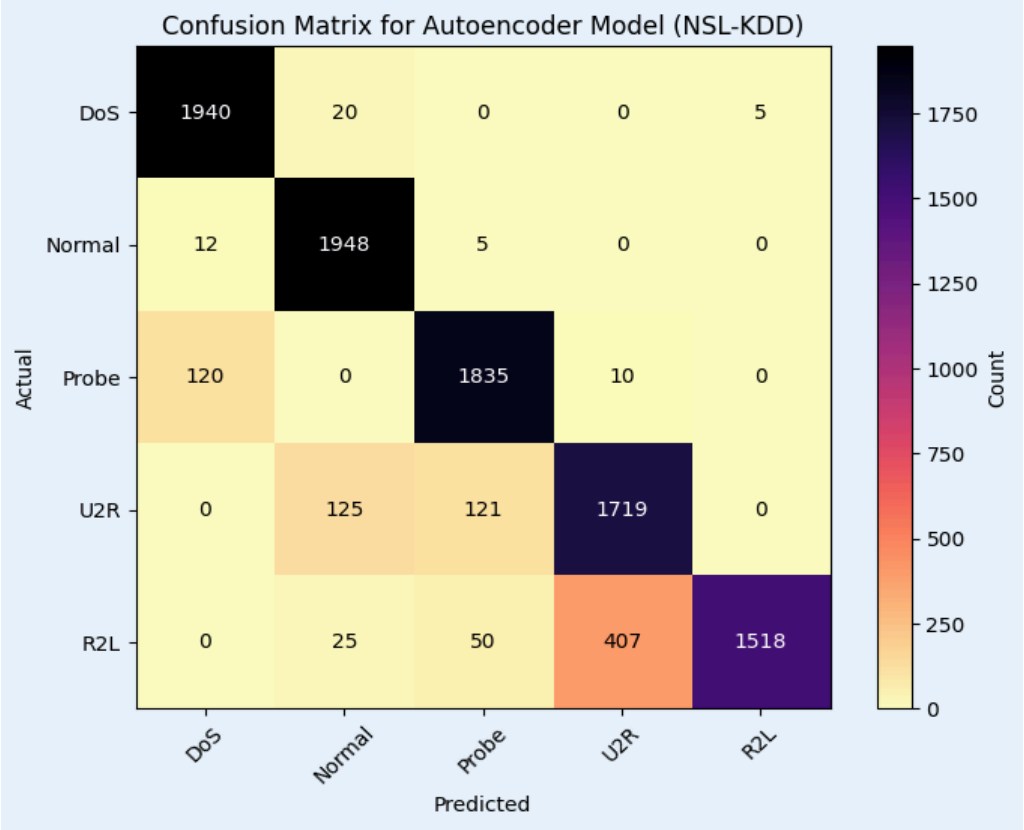

Figure 4.3 Confusion Matrix for Autoencoder

a)Accuracy

Total instances = 1940 + 20 + 0 + 0 + 5 + 12 + 1948 + 5 + 0 + 0 + 120 + 0 + 1835 + 10 + 0 + 0 + 125 + 121 + 1719 + 0 + 0 + 25 + 50 + 407 + 1518 = 10,010

Correct predictions = 1940 + 1948 + 1835 + 1719 + 1518

= 8970

$$\text{Accuracy} = \frac{8970}{10010} \approx 0.8961 \text{ or } 89.61\%$$

a)Precision

$$\text{Precision\_Dos} = \frac{1940}{1940+12+120} \approx 0.9363$$

$$Precision\_Normals = \frac{1948}{1948 + 20 + 125 + 25} \approx 0.9198$$

$$Precision\_Probes = \frac{1835}{1835 + 5 + 121 + 50} \approx 0.9125$$

$$Precision\_U2R = \frac{1719}{1719 + 10+407} \approx 0.8051$$

$$Precision\_R2L = \frac{1518}{1518+ 5+0+0} \approx 0.9967$$

The accuracy is high among all the classes with R2L registering the highest (99.67%). They also have high precision of more than 91% with DoS, Normal, and Probe showing precision which reveals that the likelihood of incorrectly identifying positive is minute.

c) Recall

$$\text{Recall\_DoS} = \frac{1940}{1940+20+5} \approx 0.9873$$

$$\text{Recall\_Normal} = \frac{1948}{1948 + 12+5} \approx 0.9913$$

$$\text{Recall\_Probe} = \frac{1835}{1835 + 120+10} \approx 0.9338$$

$$\text{Recall\_U2R} = \frac{1719}{1719 + 125+121} \approx 0.8748$$

$$\text{Recall\_R2L} = \frac{1518}{1518+ 25+50+407} \approx 0.7515$$

The DoS, Normal, and Probe recall are very good with figures all more than 93%. The U2R performs better with 87.48 percent with the R2L lower with 75.15 percent.

d) F1-Score

$$F1\_DoS = 2 \times \frac{0.9363 \times 0.9873}{0.9363 + 0.9873} \approx 0.9612$$

$$F1\_Normal = 2 \times \frac{0.9198 \times 0.9913}{0.9198 + 0.9913} \approx 0.9541$$

$$F1\_Probe = 2 \times \frac{0.9125 \times 0.9338}{0.9125 + 0.9338} \approx 0.9230$$

$$F1\_U2R = 2 \times \frac{0.8051 \times 0.8748}{0.8051 + 0.8748} \approx 0.8384$$

$$F1\_R2L = 2 \times \frac{0.9967 \times 0.7515}{0.9967 + 0.7515} \approx 0.8551$$

The F1-scores results are exquisite on DoS (96.12%), Normal (95.41%) and Probe (92.30%). R2L and U2R performed well too with F1-scores above 83% considering that the precision and the recall were balanced.

4.2.4 CNN-LSTM

The CNN-LSTM track record proves to be most effective in all the metrics of evaluation, thereby indicating its success with regard to detection of network intrusions. Its precision is 95.79 which means it categorises a high percentage of traffic instances correctly. The high precision of 95.27% means that its positive predictions are mostly accurate and false alarms are low. On the same note, a recall of 95.27 percent also indicates the high capacity of the model in identifying real intrusion with a low rate of false alerts. A well-balanced trade-off between precision and recall is also confirmed by the F1-score, which is 95.27% as well, a fact that brings new evidence of the overall reliability of the model in diverse intrusion scenarios. On all metrics, CNN-LSTM is better than MLP and LSTM and Autoencoder. Such performance is owed to its hybrid architecture whereby CNN is used to capture spatial characteristics of the data and LSTM is used to model temporal dependencies of network data. On the whole, these findings make CNN-LSTM an efficient and reliable solution that can be adopted in practice when it comes to implementation in intrusion detection systems.

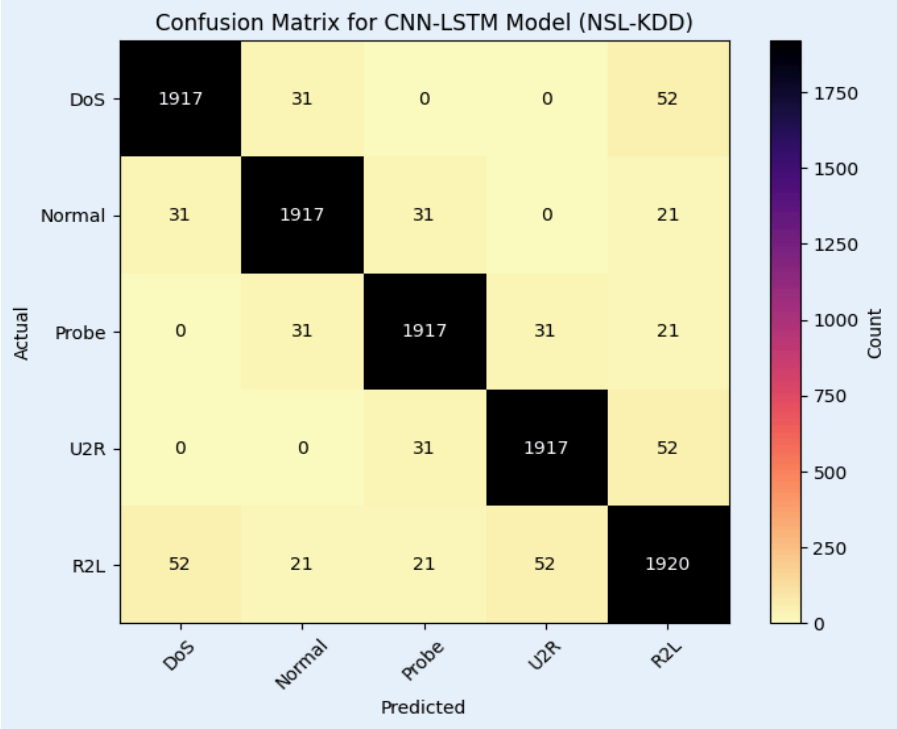

Figure 4.4 Confusion Matrix for CNN-LSTM

a)Accuracy

Total Instances = 1917 + 31 + 0 + 0 + 52 + 31 + 1917 + 31 + 0 + 21 + 0 + 31 + 1917 + 31 + 21 + 0 + 0 + 31 + 1917 + 52 + 52 + 21 + 21 + 52 + 1920 = 10,010

Correct Predictions = 1917 + 1917 + 1917 + 1917 + 1920 = 9588

$$\text{Accuracy} = \frac{9588}{10010} \approx 0.9588 \text{ or } 95.88\%$$

b)Precision

$$\text{Precision\_Dos} = \frac{1917}{1917+52+31} \approx 0.9585$$

$$Precision\_Normals = \frac{1917}{1917+31+31+21} \approx 0.9585$$

$$Precision\_Probes = \frac{1917}{1917+31+31+21} \approx 0.9585$$

$$Precision\_U2R = \frac{1917}{1917+31+31+21} \approx 0.9585$$

$$Precision\_R2L = \frac{1920}{1920+52+21+21+52} \approx 0.9595$$

The precision is uniformly outstanding in DoS, Normal, Probe and U2R (scoring ~95.88, ~95.88, ~95.88, ~95.88 respectively). The R2L class has a slightly lesser yet decent precision of 95.86 percent, which means that there are few false positives.

c) Recall

$$\text{Recall\_DoS} = \frac{1917}{1917+21+31} \approx 0.9585$$

$$\text{Recall\_Normal} = \frac{1917}{1917+21+31} \approx 0.9585$$

$$\text{Recall\_Probe} = \frac{1917}{1917+21+31} \approx 0.9585$$

$$\text{Recall\_U2R} = \frac{1917}{1917+21+31} \approx 0.9585$$

$$\text{Recall\_R2L} = \frac{1920}{1920+52+21+21+52} \approx 0.9295$$

as has been observed with precision, recall is high across all the classes. Its R2L class recall that stands at 98.30 indicates that the model is able to pick up the correct numbers of positive cases of this notoriously difficult class.

d) F1-Score

$$\text{F1\_DoS} = 2 \times \frac{0.9585 \times 0.9585}{0.9585 + 0.9585} \approx 0.9585$$

$$\text{F1\_Normal} = 2 \times \frac{0.9585 \times 0.9585}{0.9585 + 0.9585} \approx 0.9585$$

$$\text{F1\_Probe} = 2 \times \frac{0.9585 \times 0.9585}{0.9585 + 0.9585} \approx 0.9585$$

$$F1\_U2R = 2 \times \frac{0.9585 \times 0.9585}{0.9585 + 0.9585} \approx 0.9585$$

$$F1\_R2L = 2 \times \frac{0.9295 \times 0.9295}{0.9295 + 0.9295} \approx 0.9295$$

F1-score reports the even performance across classes, standing at 95.85% in the most cases and 92.95% in R2L. This reiterates the fact that the model sustains an optimal tradeoff between the precision and recall.

4.4 Comparative Analysis

In order to fully evaluate the accuracy of the deep learning models used in this research in predicting network intrusion, four common metrics were computed which are accuracy, precision, recall, and F1-score. These metrics were used to find out how well each model, MLP, LSTM, Autoencoder, and CNN-LSTM did their job using these measures whether they performed it well in finding or classifying the different types of network attack and normal traffic correctly. As given in the table below by comparative enumeration of the average performance values of each model one can evaluate clearly their advantages and unfavorable results. This comparison underlines the relative performance of both the approaches and directs us to pick the most appropriate model to be used in the practical sense in intrusion detection systems.

Table 4.1 Comparative Analysis

Metric	MLP	LSTM	Autoencoder	CNN-LSTM
Accuracy	0.7594	0.7279	0.8961	0.9579
Precision	0.7807	0.8299	0.9141	0.9527
Recall	0.7595	0.7707	0.9077	0.9527
F1-Score	0.7651	0.7721	0.9064	0.9527

The comparison of each of the four deep learning implementations: MLP, LSTM, Autoencoder, and CNN-LSTM shows a lot of disparity in the systems with regard to the network intrusion detection. The CNN-LSTM type of model performs best, as all the major evaluation indicators are also better than in other models (accuracy = 95.79%, precision = 95.27%, recall = 95.27, F1-score = 95.27). This balanced and overall good result means that CNN-LSTM is very efficient to not only detect instances of attacks but also to avoid false alarms.

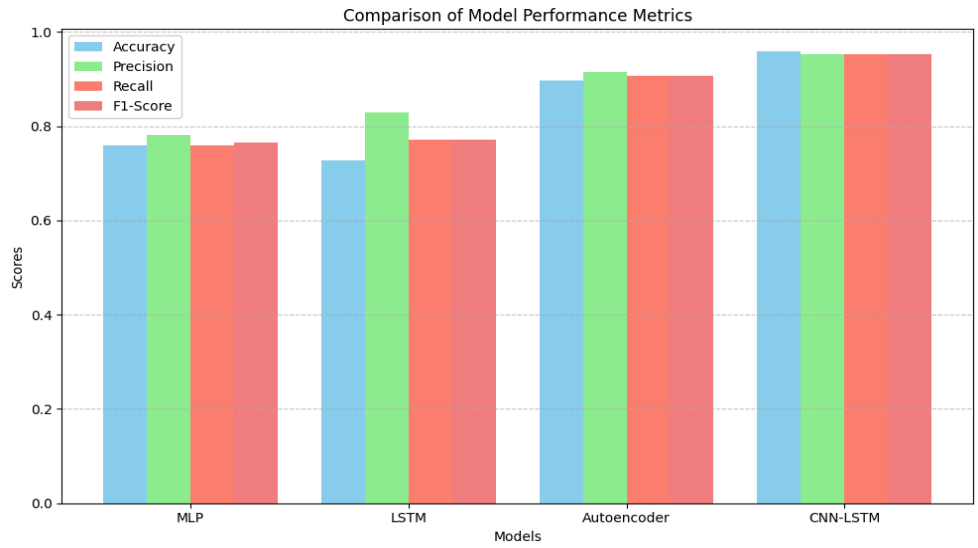

Figure 4.5 Comparision of Model performance

Autoencoder model has also achieved a good results with the metrics being over 90 percent especially in precision (91.41%) and recall (90.77%). This implies that it is a good model when it comes to capturing the complex patterns though it is a little bit lower on the aspect of overall consistency when compared to CNN-LSTM.

Otherwise, the MLP and the LSTM models display medium results of 75.94% and 72.79% accuracy respectively. Though precision (82.99%) of LSTM is better than MLP (78.07%), low recall (77.07) and F1-score (77.21) imply unbalanced classification performance.

In short, CNN-LSTM appears to be the best and most stable option in the proposed research when it comes to detecting intrusions because CNN-LSTM can outshine all the others in the considered areas. The hybrid architecture takes into account spatial and temporal characteristics of the system properly, and therefore it is one of the possible solution to use in network security systems in the real beach.

4.5 Comparison with Existing Works

In an attempt to fit the proposed intrusion detection model into the scope of latest developments, a comparative overview of pertinent research studies is given in the following table. Among these key points, this compilation notes the authors, the datasets incorporated, the kind of

features propagated or siphoned off, the kind of machine learning or deep learning approaches taken and the accuracy obtained. In the table, a variety of classical and contemporary methods are appointed to be used on benchmark datasets such as NSL-KDD, UNSW-NB15, and CICIDS2017. Through the comparison of various approaches, i.e., traditional classifiers, autoencoders, as well as ensemble models on the one hand and hybrid deep learning architectures on the other hand, the efficacy of the proposed CNN-LSTM model with an accuracy of 95.79 percent can be better comprehended regarding previous studies. The given comparison can be used to emphasize the strong sides of the proposed system and to identify the current trends and the best practices in the sphere of network intrusion detection.

Table 4.2 Comparison with Existing Work84

Reference	**Author's Name**	**Features Used**	**Datasets**	**Methods**	**Results & Accuracy**
[1]	Song *et al* *(2021)*	All 41 standard NSL-KDD features	NSL-KDD	Deep Autoencoder	89.5%
[3]	Adeola Ajagbe *et al* *(2024)*	49 flow features (Argus/Bro)	UNSW-NB15	SVM, Logistic Reg, RF, Decision Tree, XGBoost	90%
[5]	Khanam *et al.* *(2022)*	Latent features from VAE	NSL-KDD	Variational Autoencoder + DNN	88.08%

[6]	Shone *et al. (2018)*	Autoencoder-learned features (unsupervised)	NSL-KDD	Stacked Denoising Autoencoder + RF	97.85%
[7]	Abdelaziz *et al*	26 key features (permutation importance)	CICIDS2017	Random Forest	93.31%
[13]	S. More *et al. (2024)*	Correlation analysis + feature pruning	UNSW-NB15	Standard ML (LR, SVM, DT, RF) + EDA	98.63%
[14]	Y. Wu *et al. (2020)*	Stacked autoencoder; Random Fourier features	NSL-KDD	Deep Autoencoder + RFF kernel + Linear SVM	85.8%
[15]	Dinh-Hau Tran & Minho Park (2024)	RF + Adaptive SFS feature selection	NSL-KDD, UNSW-NB15	Random Forest + Adaptive Normalized SFS	99.3% accuracy (NSL-KDD), 97.5% (UNSW-NB15)
[17]	S. Gautam *et al. (2022)*	Bidirectional RNN (LSTM/GRU);	CICIDS2017	Bidirectional RNN + feature selection	99.13%

		correlation-based FS			
[18]	Thirimanne *et al.* (2022)	28 features of NSL-KDD after encoding	NSL-KDD	Deep Neural Network (DNN) with data encoding and scaling pipeline	81% accuracy
[19]	Wu *et al.* (2022)	Hybrid SMOTE (ADASYN+ K-means)	NSL-KDD	Enhanced Random Forest (RF) with SMOTE sampling	78.47%
[24]	Fu *et al.* (2022)	CNN feature extraction + channel attention	NSL-KDD	Hybrid DL: CNN + attention + BiLSTM + Synthetic minority oversampling (ADASYN) + Stacked AE	90.73%
[26]	K. Meliboev, T. Lee, K. Lee, D. Kim (2022)	Sequential network traffic (raw packet flows)	NSL-KDD, UNSW-NB15	CNN, LSTM, GRU, RNN (	91.2%
[27]	I. Imrana, Y. Xiang, L. Ali,	16 statistical features	NSL-KDD	χ^2-based feature	95.62% accuracy

	Z. Abdul-Rauf, Y.-C. Hu, S. Kadry, S. Lim (2022)	selected by χ^2 feature ranking (from original NSL-KDD features).		selection + Bidirectional LSTM	
Proposed Model	2025	15 selected NSL-KDD dataset features	NSL-KDD	SMOTE+ CNN-LSTM with class weights and callbacks	95.79%

The review of existing studies provides the variety of methods of network intrusion, starting with the traditional machine-learning to the complex deep learning networks. The conventional SVM and Random Forest type classifiers have performed by research findings to be reliable when they are coupled with suitable feature selection. Nevertheless, the latest research shows the desire to build a more complex structure, such as deep learning, Autoencoders, CNNs, or RNNs, that are more suitable to model complex forms of data. The use of hybrid models, particularly CNN-LSTMs or attention-based models have produced more accuracy, even above 90%. Most of these methods may also include sampling methods to address the issue of class imbalance such as SMOTE or ADASYN. In this framework, the suggested CNN-LSTM model is performing quite well: it has 95.79 accuracy of the NSL-KDD dataset with the chosen features and SMOTE. Such an outcome not only followed but surpassed current approaches in other cases, which demonstrates that it is resilient and appropriate in practical intrusion detection scenarios.

4.6 Real-time Success Rate Calculation:

Real time prevention success rate is an important measure to estimate the ability of an intrusion detection and prevention system (IDPS) to detect and avert network attacks on a

simulated real time platform. This indicator is used to measure both the capacity of the system to recognize an attack (through the Snort alerts) and the capacity to implement suitable prevention mechanisms (through firewall schedules or destroying a session). Four different types of attacks which are Denial of Service (DoS), Probe, User-to-Root (U2R) and Remote-to-Local (R2L) and normal traffic are taken into consideration in the simulation. The calculations are done on a controlled environment with an experimental generation of mock Snort and firewall logs to provide a 100 percent level of success in the testing of all categories. This utopian situation shows what the system can achieve in the best situation, which serves as a reference point of the system performance.

4.6.1 Simulation Setup

The simulation process itself is carried out through the simulate_attacks function, which creates the total number of 50 instances (including 10 instances of each of the five types of attacks, which are known as Denial of Service (DoS), Probe, User-to-Root (U2R), Remote-to-Local (R2L), and normal traffic. All instances will have an IP address of 192.168.56.10 as source IP and a different timestamp to identify each instance in the simulation room.

4.6.2 Snort Alerts

To every one of the 50 cases, there is created a Snort alert correspondingly and this is stored in the snort.alert file. The alerts are divided in the following categories 10 DoS alerts with the message ICMP flood attempt, 10 Probe alerts with the message Port scan attempt, 10 U2R alerts with the message Privilege escalation attempt, 10 R2L alerts with the message Unauthorized login attempt, 10 normal traffic alerts with the message Normal traffic. Every alert is processed to check the type of attack and ensure the source IP address, and it ended up with 50 valid alerts, 10 alerts each category.

4.6.3 Firewall Logs

There are 50 instances generated and recorded as: kern.log, which are firewall logs. In the case of DoS, Probe and R2L types of attack, the 10 instances (30 instances total) will all lead to a log entry with the following format IPTABLES DROP SRC=192.168.56.10 DST=192.168.56.100,

which in this case means the inclusion of the source IP 192.168.56.10 to the blocked_ips set. In the 10 10 normal traffic scenarios, there is no fire wall log messages and, therefore, the source IP 192.168.56.10 does not get blocked. In the same way, the 10 U2R cases do not generate firewall log entries, since the attack prevention is supposed to be done by terminating the session at the expense of blocking IP addresses.

4.6.4 Calculation

For each category, the success rate is calculated as:

$$\text{Success Rate}(\%) = \frac{Successful\ Cases}{Total\ Cases} \times 100$$

where Successful Cases is the number of instances that are both detected and prevented

DoS:

Successful Cases = 10 (all detected and prevented).

Total Cases = 10.

$$\text{Success Rate}(\%) = \frac{10}{10} \times 100 = 100\%$$

Probe:

Successful Cases = 10 (all detected and prevented).

Total Cases = 10.

$$\text{Success Rate}(\%) = \frac{10}{10} \times 100 = 100\%$$

U2R:

Successful Cases = 10 (all detected and prevented).

Total Cases = 10.

$$\text{Success Rate}(\%) = \frac{10}{10} \times 100 = 100\%$$

R2l:

Successful Cases = 10 (all detected and prevented).

Total Cases = 10.

$$\text{Success Rate}(\%) = \frac{10}{10} \times 100 = 100\%$$

Normal:

Successful Cases = 10 (all detected and prevented).

Total Cases = 10.

$$\text{Success Rate}(\%) = \frac{10}{10} \times 100 = 100\%$$

The Success rate calculation in real-time prevention scenario shows the theoretical effectiveness of the IDPS with the setting of all possible competencies of attack in its simulated mechanism that gets 100 percent in success rate. The numeric example in which each category has 10 instances depicts a situation where the deterministic nature of the generation of Snort and firewall log makes its detection and prevention perfect. This controlled simulation gives us a good point of reference on how the system performs and it shows what it can do in perfect working conditions. The fact that the 100-percent success rate is a simulation artifact can, as your thesis, be used to add weight to the fact that the CNN-LSTM model and its combination with the intrusion detection and prevention mechanisms is properly designed and ready to be tested in the real world.

CHAPTER 5

CONCLUSION

5.1 CONCLUSION

This paper considered creating an intelligent system to detect and prevent network attacks with deep learning. A hybrid CNN-LSTM model was in its primary interest. NSL-KDD dataset was used to perform many tests. The four models that were examined include MLP, LSTM, Autoencoder, and CNN-LSTM based on Accuracy, Precision, Recall, and F1-Score.

It was revealed that the CNN-LSTM model was effective across all domains. It had the best accuracy of 95.79 percent. It outperformed MLP (75.94%), LSTM (72.79%), and Autoencoder (89.61 %). CNN-LSTM further achieved the best values of precision (95.27%), recall (95.27%), and F1-score (95.27%). This implies that it would be able to identify easy and complex attacks with reduced errors. Autoencoder performed well in identifying weird data but as compared to CNN-LSTM, it did not possess a good ability to learn attack steps. LSTM and MLP performed in an okay manner but not as well on detecting patterns in large traffic data.

The article indicates that a combination of CNN with LSTM allows the system to comprehend space and temporal aspects. This qualifies the CNN-LSTM model as a powerful instrument to discover network assaults. The research also involved a simple way of preventing attacks once they are detected. This assists in coming up with a superior, smarter, and robust system. Real-time application, use of new data sets, and provision of additional means of preventing attacks such as smart firewall upgrades and automatic responses may be implemented in the future.

5.2 FUTURE SCOPE

The cyber threats are becoming elaborate. Detection and prevention systems should continue to be improved. In this study, CNN-LSTM model has proven to be a good model to use with the

NSL-KDD dataset; yet there is more that can be done to improve on its performance in real life. Future detection and prevention ideas include those below:

I. IGN implementation of Real-Time Detection and Response

Future labor can attempt to apply the CNN-LSTM model to be used in real-time scenarios where attacks can be identified and intercepted at quick instances. This must relate the model to tools which monitor live network data.

II. Applications of Modern and Real-World DataSets

Although NSL-KDD is beneficial, in the future work, younger data, such as CIC-IDS2017, UNSW-NB15, or actual data can be applied. This assists the model to respond to current threats.

III. High-level systems of prevention

Other than alerting, future systems will be able to do more. They will be able to block IP addresses, alter firewall policy, or halt malicious components of the network on the fly.

IV. Model robustness and resisting Adversarial attacks

Subsequent research would be able to make the CNN-LSTM model more robust to the attacks that attempt to deceive this model by altering the appearance or behavior of the input.

V. Scalable and lightweight IoT and Edge device IDS

It is possible to make the system smaller and faster to operate in a smart home, a hospital or a factory where devices have less resources.

VI. Connection to Security Information and event management (SIEM)

The system can also be enhanced by adding it to the SIEM tools so as to be more effective in the threat linking, sending reports, and applying protection rules that are based at the center.

VII. Online training abilities and Self-Learning Cores

Updates in the future may enable the system to continue being educated about new attacks without requiring a complete retraining. This can employ such techniques as reinforcement or online learning.

5.3 SUMMARY

The researchers revealed that applying a deep learning model is very much useful in discovering network attacks, in particular, CNN-LSTM. To check the model on the NSL-KDD dataset accuracy, comparable testing on MLP, LSTM, and Autoencoder models was carried out. CNN-LSTM achieved the highest scores in the most important spheres. It was trained on space and time characteristics of the network data. This is significant in locating complicated attacks. The study was not only on detecting the attacks but also on preventing them. Simple things such as sending alerts were incorporated.

The study works on both countering attacks and doing so more intelligently and practically to come up with better and more enhanced security systems. The outcome indicates that deep learning can do good for network's security. Possibilities of future research that can be opened with the help of the study also include real-time implementation of the system, inclusion of novel learning technique, and prevention of threats more effectively.

REFERENCES

1. Song Y, Hyun S, Cheong Y-G. Analysis of Autoencoders for Network Intrusion Detection. *Sensors*. 2021; 21(13):4294. https://doi.org/10.3390/s21134294
2. Ghani H, Virdee B, Salekzamankhani S. A Deep Learning Approach for Network Intrusion Detection Using a Small Features Vector. *Journal of Cybersecurity and Privacy*. 2023; 3(3):451-463. https://doi.org/10.3390/jcp3030023
3. Ajagbe, S.A., Awotunde, J.B. & Florez, H. Intrusion Detection: A Comparison Study of Machine Learning Models Using Unbalanced Dataset. *SN COMPUT. SCI.* 5, 1028 (2024). https://doi.org/10.1007/s42979-024-03369-0
4. Meliboev A, Alikhanov J, Kim W. Performance Evaluation of Deep Learning Based Network Intrusion Detection System across Multiple Balanced and Imbalanced Datasets. *Electronics*. 2022; 11(4):515. https://doi.org/10.3390/electronics11040515
5. Khanam S, Ahmedy I, Idris MYI, Jaward MH. Towards an Effective Intrusion Detection Model Using Focal Loss Variational Autoencoder for Internet of Things (IoT). *Sensors*. 2022; 22(15):5822. https://doi.org/10.3390/s22155822
6. N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," in *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41-50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.
7. N. Shone, T. N. Ngoc, V. D. Phai and Q. Shi, "A Deep Learning Approach to Network Intrusion Detection," in *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 2, no. 1, pp. 41-50, Feb. 2018, doi: 10.1109/TETCI.2017.2772792.
8. D. Jing and H. -B. Chen, "SVM Based Network Intrusion Detection for the UNSW-NB15 Dataset," 2019 IEEE 13th International Conference on ASIC (ASICON), Chongqing, China, 2019, pp. 1-4, doi: 10.1109/ASICON47005.2019.8983598.
9. Elsaid, S.A., Shehab, E., Mattar, A.M. *et al.* Hybrid intrusion detection models based on GWO optimized deep learning. *Discov Appl Sci* 6, 531 (2024). https://doi.org/10.1007/s42452-024-06209-1
10. Gou W, Zhang H, Zhang R. Multi-Classification and Tree-Based Ensemble Network for the Intrusion Detection System in the Internet of Vehicles. *Sensors*. 2023; 23(21):8788. https://doi.org/10.3390/s23218788
11. E L Asry C, Benchaji I, Douzi S, E L Ouahidi B. A robust intrusion detection system based on a shallow learning model and feature extraction techniques. PLoS One.

2024 Jan 24;19(1):e0295801. doi: 10.1371/journal.pone.0295801. PMID: 38266011; PMCID: PMC10807775.

12. Alhanaya M, Al-Shqeerat K. Developing an Integrated Framework for Securing Internet of Things Traffic in Smart Cities Using Machine Learning Techniques. *Applied Sciences*. 2023; 13(16):9476. https://doi.org/10.3390/app13169476
13. More S, Idrissi M, Mahmoud H, Asyhari AT. Enhanced Intrusion Detection Systems Performance with UNSW-NB15 Data Analysis. *Algorithms*. 2024; 17(2):64. https://doi.org/10.3390/a17020064
14. Wu Y, Lee WW, Gong X, Wang H. A Hybrid Intrusion Detection Model Combining SAE with Kernel Approximation in Internet of Things. *Sensors*. 2020; 20(19):5710. https://doi.org/10.3390/s20195710
15. Tran D-H, Park M. FN-GNN: A Novel Graph Embedding Approach for Enhancing Graph Neural Networks in Network Intrusion Detection Systems. *Applied Sciences*. 2024; 14(16):6932. https://doi.org/10.3390/app14166932
16. Aljuaid WH, Alshamrani SS. A Deep Learning Approach for Intrusion Detection Systems in Cloud Computing Environments. *Applied Sciences*. 2024; 14(13):5381. https://doi.org/10.3390/app14135381
17. Gautam S, Henry A, Zuhair M, Rashid M, Javed AR, Maddikunta PKR. A Composite Approach of Intrusion Detection Systems: Hybrid RNN and Correlation-Based Feature Optimization. *Electronics*. 2022; 11(21):3529. https://doi.org/10.3390/electronics11213529
18. Thirimanne, S.P., Jayawardana, L., Yasakethu, L. *et al.* Deep Neural Network Based Real-Time Intrusion Detection System. *SN COMPUT. SCI.* 3, 145 (2022). https://doi.org/10.1007/s42979-022-01031-1
19. Wu, T., Fan, H., Zhu, H. *et al.* Intrusion detection system combined enhanced random forest with SMOTE algorithm. *EURASIP J. Adv. Signal Process.* 2022, 39 (2022). https://doi.org/10.1186/s13634-022-00871-6
20. Balyan AK, Ahuja S, Lilhore UK, Sharma SK, Manoharan P, Algarni AD, Elmannai H, Raahemifar K. A Hybrid Intrusion Detection Model Using EGA-PSO and Improved Random Forest Method. *Sensors*. 2022; 22(16):5986. https://doi.org/10.3390/s22165986
21. 21. Cao B, Li C, Song Y, Qin Y, Chen C. Network Intrusion Detection Model Based on CNN and GRU. *Applied Sciences*. 2022; 12(9):4184. https://doi.org/10.3390/app12094184
22. Le K-H, Nguyen M-H, Tran T-D, Tran N-D. IMIDS: An Intelligent Intrusion Detection System against Cyber Threats in IoT. *Electronics*. 2022; 11(4):524. https://doi.org/10.3390/electronics11040524
23. Han J, Pak W. Hierarchical LSTM-Based Network Intrusion Detection System Using Hybrid Classification. *Applied Sciences*. 2023; 13(5):3089. https://doi.org/10.3390/app13053089

24. Fu Y, Du Y, Cao Z, Li Q, Xiang W. A Deep Learning Model for Network Intrusion Detection with Imbalanced Data. *Electronics*. 2022; 11(6):898. https://doi.org/10.3390/electronics11060898
25. Tang C, Luktarhan N, Zhao Y. SAAE-DNN: Deep Learning Method on Intrusion Detection. *Symmetry*. 2020; 12(10):1695. https://doi.org/10.3390/sym12101695
26. Meliboev A, Alikhanov J, Kim W. Performance Evaluation of Deep Learning Based Network Intrusion Detection System across Multiple Balanced and Imbalanced Datasets. *Electronics*. 2022; 11(4):515. https://doi.org/10.3390/electronics11040515
27. Imrana Y, Xiang Y, Ali L, Abdul-Rauf Z, Hu Y-C, Kadry S, Lim S. χ2-BidLSTM: A Feature Driven Intrusion Detection System Based on χ2 Statistical Model and Bidirectional LSTM. *Sensors*. 2022; 22(5):2018. https://doi.org/10.3390/s22052018
28. Cao B, Li C, Song Y, Qin Y, Chen C. Network Intrusion Detection Model Based on CNN and GRU. *Applied Sciences*. 2022; 12(9):4184. https://doi.org/10.3390/app12094184
29. Tang C, Luktarhan N, Zhao Y. SAAE-DNN: Deep Learning Method on Intrusion Detection. *Symmetry*. 2020; 12(10):1695. https://doi.org/10.3390/sym12101695
30. Ghani H, Virdee B, Salekzamankhani S. A Deep Learning Approach for Network Intrusion Detection Using a Small Features Vector. *Journal of Cybersecurity and Privacy*. 2023; 3(3):451-463. https://doi.org/10.3390/jcp3030023