Abstract

Nowadays, practically every facet of modern life uses internet-based services. An exponential rise in the number of internet-connected devices and cyber-attacks has been seen in the last decade. Distributed Denial of Service (DDoS) attack is one of the most dangerous hazards on the cyber space today, affecting the availability of vital services. As a result, DDoS attack detection is the profound importance and has become a hot topic in research community. Existing DDoS detection models based on machine learning and deep learning do not focus on the computational cost of the models, especially cost of time. The models are futile, if they take much time for the detection. LIDS: A Light-weight Intrusion Detection System model and an effective feature selection method is presented in this paper. The proposed model is trained and tested on the CICDDoS2019 dataset. Our model can make the prediction in less than 1.6 microseconds which is 6x faster than the best models in literature. The model predicts with a performance of more than 99 % (accuracy, recall, precision and f1 score).

Keywords: Cyber Security, DDoS Detection, CNN, Feature Engineering, CICDDoS2019.

1. Introduction

In today's fast-paced world, it's difficult to imagine life without the Internet, which is used in several fields, including communication, education, e-commerce and so on [1]. Everything from the business field to educational sector has transitioned from offline to online during the COVID-19 epidemic. The tremendous expansion of connected devices, artificial intelligence, and the evolution of big data all point to a future that is increasingly linked and technologically based [2],[3]. These linked gadgets' services and operations are continuing to revolutionise essential industries including healthcare, transportation, residences, and ordinary people's lives. As a result, the number of intrusions and assaults employing Internet-based technologies has risen tremendously [1]. It is anticipated that the number of devices connected to the Internet of Things (IoT) would reach 38.6 billion by 2025 and 50 billion by 2030 [4]. The emergence of these gadgets, as well as the vulnerabilities that come with it, has created a unique attack surface that has attracted cyber-attackers who want to take control of them and use them to launch assaults on vital infrastructure [4]. Among several network security issues, Distributed Denial of service attack (DDoS) is one of the serious and hazardous to the network.

DDoS attacks aim to interrupt services by sending data packets that are larger than the system's ability to handle. To transmit enormous volumes of packages, attackers employ zombie machines generated with malicious software installed on victims' systems. DDoS assaults generate a surge in network traffic as packets are broadcast over the network, preventing normal users from receiving services [5].

The number of DDoS attacks launched from 2018-2022 is shown in figure 1 [6]. The attacks since 2018 have increased from 7.9 million to 13.9 million in 5 years . The trend line plotted for the data shows that the attacks will skyrocket in the coming years. The attacks will increase at the rate of 46% approximately every year.

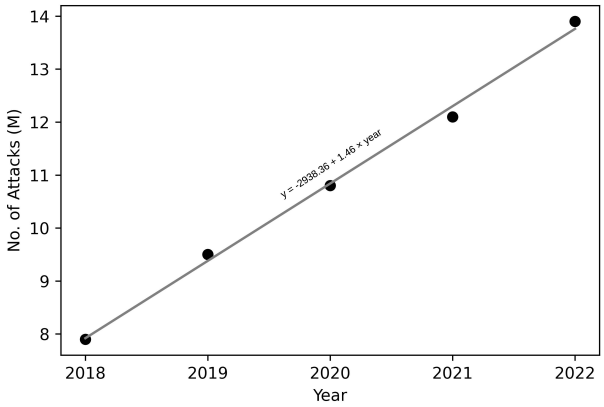

Figure 1: Number of attacks per annum

The leading target of attacks continues to be the United States (54 percent). India had a substantial spike in assaults, rising from 2% to 23% from ist half of 2021 to second half of 2021 [7]. On August 10, 2021, there were 4,296 attacks—the most ever recorded in a single day [7] . Microsoft neutralized approximately 359,713 distinct threats on our worldwide network during the 2nd quarter of 2021, a 43 % rise over the Ist quarter of 2021 [7]. As a result, research on DDoS attack detection has long been a hot area in the academic and business world.

The aim of this article is to develop a deep learning-based light weight intrusion detection model that can efficiently detect whether the incoming traffic is normal or DDoS based attack traffic, in order to prevent the halting of services. The following is the brief summary of the contribution of this paper:

- An efficient feature selection method for the training of DDoS model.
- A light weight intrusion detection model that predicts the packet to be normal or attack in less than 1.6 micro seconds which is 6 times faster than the best models in the literature.
- A model with more than 99% performance (Recall, Precision, Accuracy and F1 Score) as well.
- In order to demonstrate that the suggested model is successful, the evaluation results of the proposed model has been compared with the results from the existing approaches and the relevant literature.

The rest of the paper is arranged as follows: Section 2 discusses and reviews the related work . Section 3 discusses the proposed methodology with respect to prepossessing operations and model architecture. Section 4 describes the experimental analysis and results. Finally, the conclusion is presented in section 5.

2. Related Work

2.1. Literature Review Methodology

A systematic strategy is followed to examine the state-of-the-art methods and various approaches proposed by various researchers for DDoS detection. The literature review methodology is shown in figure 2.

Since most of the famous digital libraries such as IEEE Explore, Springer, and Science Direct are indexed by google scholar, we decided to examine the results given by the google scholar using the search string "DDoS Detection using Machine learning and Deep Learning ". The results were filtered using the date from 2020-2022. From the vast amount the publications we decided to investigate 54 papers based on the title. Upon further investigation of abstracts and conclusion 30 papers are selected for our literature review.

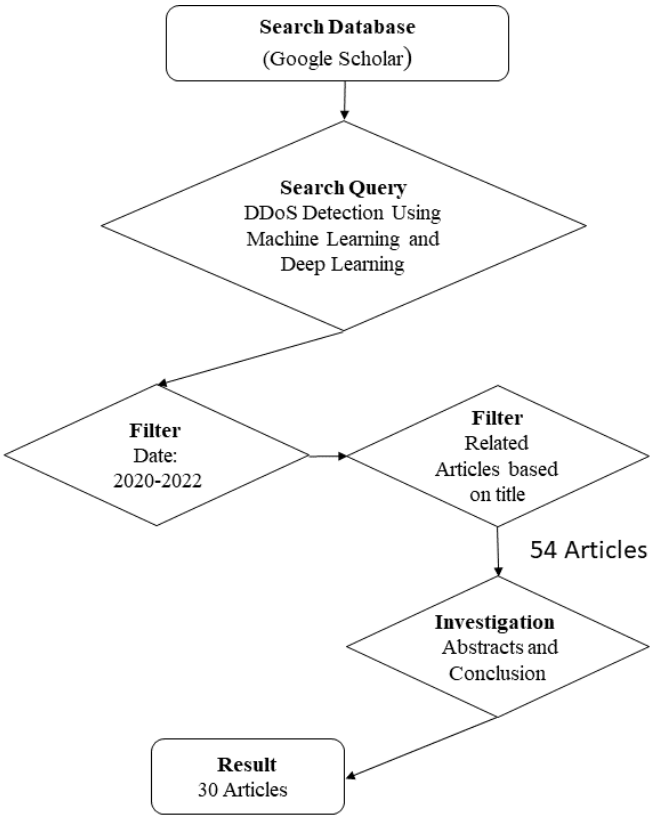

Figure 2: Flowchart of Literature Review

2.2. Literature Review

This section presents an overview of research using machine learning and deep learning models on DDoS attack detection. The machine learning and deep learning models, datasets, methodologies, and outcomes are listed in Tables 1 and 2, respectively.

Ismail *et al* [8], proposed a framework based on machine learning for the categorization and prediction of DDoS attacks using UNSW-NB 15 dataset. From the experimental results, XGBoost classifier predicts with an accuracy of 90%, outperforms the random forest classifier by 1%.

Najar *et al* [9] conducted experiments with Random Forest, K-Nearest Neighbour (KNN), Support Vector Machine (SVM) and Multi-Layer Perceptron (MLP) using NSL-KDD dataset.From the experimental results, it has been observed that, RF outperformed the other models by predicting with an accuracy of 97%.

Akshat Gaurav *et al* [10] use statistical and machine learning based methods for Intrusion Detection System (IDS). The experiments were performed using Gradient boosting (GBC), Logistic regression (LR),Decision Tree Classifier(DTC),RF and SVM models. Out of these models, GBC outperforms with an accuracy of 92.8% .

Raj kumar *et al* [11] proposed a detection model based on extreme learning machine (ELM) trained on CICDDoS2019 dataset. The ELM classifier uses multiple weight ranges, hidden neurons, and activation functions to classify the attacks based on the data obtained. The results of the experiments show that the proposed model has a detection accuracy of 99.94%, which is superior than earlier methods.

An innovative proactive feature selection approach (PFS) using DNS queries is proposed by Nuiaa *et al* [12] for early DRDoS threat identification. The PFS model was tested on CICDDoS2019 dataset. The PFS model is performing with an accuracy 0f 91.43% based on experimental data.

Raj kumar Batchu *et al* [13] suggest a new automated detection strategy that reduces overfitting and speeds up model construction by reducing feature space. Each study uses the CICDDoS2019 dataset. The GB model's 99.97% accuracy beats state-of-the-art approaches.

Khoei *et al* [14] and Alamri *et al* [15] developed ensemble-based techniques by integrating a variety of bagging, boosting, and stacking algorithms. Results show that several of these approaches outperform Random Forest, Nave Bayes (NB), and

Table 1: The Summary of existing Machine Learning-based approaches

Paper	**Year**	**Model**	**Dataset Used**	**Performance (Accuracy%)**
Ismail et al [8]	2022	XGBoost	UNSW NB 15	90
Najar et al [9]	2022	RF	NSL KDD	97
Akshat Gaurav et al [10]	2022	LR	Generated Dataset	92.8
Raj Kumar Batchu et al [11]	2022	ELM	CICDDoS2019	99.94
Nuiaa et al [12]	2022	PFS	CICIDS2017	91.4
Raj Kumar Batchu et al [13]	2021	GB	CICDDoS 2019	99.97
Khoei et al [14]	2021	Stacking, Bagging, Boosting	CICIDS2019	92.2
Pontes et al [16]	2021	EFC	CICDDoS2019	98.1
Varghese et al [17]	2021	D3	CICDDoS2019	96.58
Maseer et al [18]	2021	KNN, NB, RE, SVM	CICDDoS2017	98.86
Gohil et al [19]	2020	DT, NB, LR, SVM, KNN	CICDDoS2019	97.72
Alamri et al [15]	2020	XGBoost	CICDDoS2019	99.9

KNN in recognising DDoS attacks in diverse application contexts (e.g., Smart Grid, IoT).

An energy-based flow classifier (EFC) devised by Pontes *et al* [16] can be used to predict anomaly scores from samples that have been judged to be benign. This information is then utilised to categorise different DDoS assaults. the model classifies with a F1 score of 97.5%

Varghese *et al* [17] suggested a statistical anomaly detection approach to identify DDoS assaults in near real-time as component of an IDS'S in the data plane of a Software Defined Network (SDN). Based on experimental findings, the DPDK based DDoS Detection (D3) model is performing with an accuracy 0f 96.58% and f1 score of 0.90% on CICDDoS2019 dataset%.

Masheer *et al* [18] and Ghoil *et al* [19] employed 10 supervised and unsupervised machine learning algorithms on several DDoS datasets. The CICDDoS2017 dataset is considered as a benchmark dataset for this study. From the experimental results, the machine learning anomaly-based IDS (ML-AIDS) algorithm outperforms the other unsupervised machine learning algorithms with the accuracy of 99.28%.

Deep learning models such as Deep Neural Network (DNN), Convolutional Neural Network (CNN), and Long short-term memory (LSTM) were used by Devrim Akgun *et al* [20] to develop an IDS. The model proposed is based on inception architecture and the best results were reported. The model performs the prediction at an accuracy of 99.9%. The experimental results show that the CNN-based inception-like model gave the best results among the suggested models, with 99.9 % accuracy in binary classification and 99.3 % accuracy in multi-class accuracy. The suggested model is based on different numbers of Conv1d layers found via testing. Also, the authors added two inception-like blocks with three Conv1d layers to improve network performance.

Amaizu *et al* [21] used deep learning to detect DDoS in fifth generation (5G) and beyond 5G (B5G). This system combines two DNN models and Pearson Correlation Coefficient (PCC) feature extraction with a input layer, 7 hidden layers, two dropout layers and output layer. The suggested system was tested on CICDDoS2019 dataset. From the experimental results, the proposed method identified DDoS attacks with accuracy of 99.66% and loss score of 0.011.

Cil *et al* [5] proposed an IDS based on CNN with 3 hidden layers and 50 neurons in each layer. The model was trained and tested on CICDDoS2019 dataset. Accuracy of 99.97% and recall of 99.98% is reported by the author.

Assis *et al* [22] proposed a defence system against DDoS and intrusion threats in an SDN environment. There are two main components to this proposed system: detection and prevention. They tested their model against 7 various machine learning algorithms on the CICDDoS 2019 and CICIDS 2018 datasets. Results showed that Gated Recurrent Units (GRU) deep learning (DL) method was able to identify DDoS and intrusion assaults with an accuracy of 97.09% in all of the test conditions.

Wei *et al* [23] proposed a hybrid model named AE-MLP consisting of an Auto-encoder (AE) and a Multi-layer Perceptron Network (MLP) for effective DDoS attack detection and classification. The authors have tested AE architectures on CICDDoS2019 dataset with varied inputs, hidden layers, and hyperparameters. The optimal AE design includes 77 encoded features as input, a single hidden layer with 32 neurons, and 24 features as the final hidden layer. Accuracy of 98.34% and recall of 98.48% is reported by the author.

Deep learning intrusion detection methods were proposed by Ferrag *et al* [24] in the context of Agriculture 4.0. Models with DNN, CNN and recurrent neural networks (RNN) architectures were evaluated using binary and multi-class classifications. A

Table 2: The Summary of existing Deep Learning-based approaches

Paper	Year	Model	Dataset Used	Performance (Accuracy%)
Devrim et al [20]	2022	CNN	CICDDoS 2019	Accuracy 99.99
Amaizu et al [21]	2021	DNN	CICDDoS 2019	Accuracy 99.66
Cil et al [5]	2021	DNN	CICDDoS 2019	Accuracy 99.97
Macrcos V.O et al [22]	2021	GRU	CICDDoS 2019	Accuracy 95.4
Yianyuan Wei et al [23]	2021	AE-MLP	CICDDoS 2019	Accuracy 98.34
Ferrag et al [24]	2021	CNN,RNN,DNN	CICDDoS 2019	Accuracy 95.12,94.88,93.88
Chin Shiuh et al [25]	2021	Bi-LSTM	CIC-IDS 2017 and CICDDoS 2019	Accuracy 94
Shieh et al [25]	2021	Bi-directionalLSTM + Gaussian Mixture	CICDDoS 2019	Accuracy 98
Sanchez et al [26]	2021	MLP	CICDDoS 2019	Accuracy 99.93
Samom and Taggu et al [27]	2021	MLP	CICDDoS 2019	Accuracy 99.92
Can et al [28]	2021	MLP	CICDDoS 2019	F1 Score 79.39
Elsayed et al [29]	2020	RNN+ Autoencoder	CICDDoS 2019	F1 Score 99
Almaini et al [30]	2021	KAlman Back propogation NN	CICDDoS 2019	Accuracy 94
Hussain et al [31]	2020	CNN (ResNet)	CICDDoS 2019	Precision 87
Asad et al [32]	2020	DNN	CICIDS 2017	Accuracy 98
Shurman et al [33]	2020	LSTM	CICDDoS 2019	Accuracy 99.19
Jia et al [34]	2020	LSTM	CIC-IDS 2017 and CICDDoS 2019	Accuracy 98.9
de Assis et al [35]	2020	CNN	CIC-IDS 2018 and CICDDoS 2019	Accuracy 95.4

variety of DDoS assaults were practised on their models with the help of TON IoT and CIC-DDoS2019.

Chin Shiuh *et al* [25] present a DDoS attack detection architecture using a Gaussian Mixture Model (GMM), incremental learning, and Bidirectional (BI-LSTM) with 2 bidirectional layers and a learning rate of 0.00859. It has been shown via experimentation that the proposed BI LSTM-GMM is capable of achieving recall, precision, and accuracy of 94%.

Shieh *et al* [25] deployed a Bi-directional LSTM model with a GMM to identify and classify six different types of DDoS assaults with a 98% accuracy. The CICDDoS 2019 dataset is being utilised for the purpose of evaluation.

Sanchez *et al* [26]presented a standalone Multi-Layer Perception (MLP) with an accuracy of 99.93 percent and a 99.96 percent F1-score.

Samom and Taggu [27] proposed an MLP model and compared the results with other machine learning algorithms, to recognise four distinct DDoS assaults (SYN, NET, Portmap, and UDPLag). They employ the principal chi-squared function to choose 20 characteristics and the principal component analysis (PCA) technique to reduce the dimensions in their study. On the CICDDoS2019 dataset, their submission showed that their model had a 99.92% accuracy.

Can *et al* [28] proposed DDoSNet, which classifies features using a fully linked MLP and automated classification technique (Feature Selection). This method reported 91.16% accuracy, 79.41% recall, and 79.39% F1-score for multi-class classification. The authors underlined that their methodology had low performance since they used the entire feature set for categorization, which was a drawback of their method.

Elsayed *et al* [29] introduced a hybrid solution dubbed DDoSNet for identifying DDoS assaults at the Software-Defined Networking (SDN) layer by combining a Recurrent Neural Network (RNN) with an Auto-encoder. The analysis of their proposal found that the DDoSNet model had the best performance metrics according to the Confusion Matrix, despite the fact that they only give binary classification.

The Kalman Back-propagation Neural Network was developed by Almaini *et al* [30] using the Kalman approach to modify weight metrics and back-propagation to correct biases. In terms of accuracy, their model was 94% accurate with a false alarm rate of only 0.0952.

Hussain *et al* [31] devised a technique for detecting DoS and DDoS attacks on IoT devices using the ResNet18 deep learning architecture. They used the obtained dataset to train Residual Networks (ResNet18) on the properties of network traffic data. The deep learning model was trained to distinguish between 11 distinct types of assaults and legitimate traffic.

Asad *et al* [32] proposed a deep detect model, based on DNN architecture. The model was trained on CICDDoS2017 dataset and predicts with an F1 score of 0.99%

Shurman *et al* [33] proposed a hybrid and an LSTM based IDS model. The model was trained on reflection based CICDDoS2017 dataset. LSTM based model outperforms hybrid model with an accuracy of 99.19%.

Jia *et al* [34] proposed convolutional and LSTM-based models for detecting DDoS attacks on IoT networks based on changes in traffic. CIC-DDoS2019 data was combined with

the DDoS simulator datasets generated by Botnet Simulator (BoNeSi) and SlowHTTPPTest. For IoT networks, they analysed the model's suitability and evaluated its performance.

DDoS assaults can be mitigated using a software-defined network (SDN) security solution proposed by De Assis *et al* [35] for the Internet of Things (IoT). To classify assaults, they looked at machine learning models like MLP and CNN as well as Dense-MLP and LR. In addition to generating SDN traffic for training and testing, the CIC-DDoS2019 data set was used to collect simulated IP flows.

2.3. Dataset Description (CICDDoS2019)

This study used the CIC-DDoS2019 dataset [36], the latest version provided by the Canadian Institute for Cyber security, they has been providing datasets on cyber attacks and undertaking statistical research on them for years. This dataset was collected by the authors via the use of CICFlowMeter-V3 in real-world settings. The dataset was produced between 9:43 a.m. and 17:35 p.m. A total of 12 kinds of DDoS attack types as shown in Figure 3 were reported between 10:35 a.m. and 17:15 a.m. on the second day of the assault campaign. PCAP files or comma-separated values (CSV) files may be used to retrieve this data. A total of 11 CSV files are provided in the dataset that contain a total of 13 labels constituting 12 types of attack packets and one normal packet. These attacks are broadly categorized into different categories of attacks as shown in figure 3.

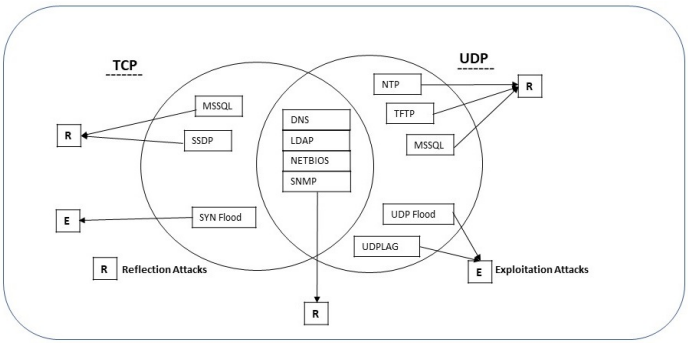

Figure 3: Taxonomy of DDoS attacks based on protocol

3. Proposed Methodology

The proposed solution has two stages, pre-processing stage which makes the data ready for the model training and the second stage is model architecture (discussed in section 3.2) trained on the cleaned dataset that is obtained in stage 1.

*3.1. **Pre-processing***

Figure 4 shows the pre-processing operations to obtain the dataset to be used in the training of our IDS model. There are 11 files and 12 attack types and 1 normal present in the CICDDoS2019 dataset with a total of 5,00,63,112 records and 88 features.

3.1.1. Data Cleaning

The files are processed and cleaned using Algorithm 1 and Algorithm 2. The various modules of the data cleaning algorithm 1 and 2 are explained in the sections below.

Algorithm 1 : Data Cleaning 1

Step 1: dataset := F_i, { i = 1, $F_i \in Files$}
Step 2: dataset := drop_meaningless_features (dataset)
Step 3: dataset := drop_null_values (dataset)
Step 4: dataset := drop_duplicates (dataset)
Step 5: $\forall$ F_i in F { i = 2, 3, ..., N}
    data =: *repeat* steps 1-4
    dataset := concatnate [data, dataset]
    dataset := drop_duplicates()

- *Dropping Meaningless Features:*

Step 2 of algorithm 1 deletes the meaningless features in the data file. There are eight meaningless features which make no sense to use for our IDS model. These are *Unnamed: 0, Flow ID, Timestamp, Source IP, Source Port, Destination IP, Destination Port, SimillarHTTP*. The step is applied to each data file, deleting these meaningless features. After this, the dataset is left with a total of 80 features.

- *Dropping Null Values:*

Step 3 of algorithm 1 deletes the records having any null values. As the data files are huge and the data at missing is missing at random (MAR) the list-wise deletion of null values gives the same results as any other imputation method, like maximum likelihood or multiplr imputation etc. So the removal technique was used to be keeping in mind the memory efficiency.

- *Dropping duplicate records:*

Step 4 of algorithm 1 drops any duplicate records in the data file, the redundant records do not contribute to any other additional information, so they are deleted.

Each file is loaded and processed using steps 2 – 4 as mentioned in the step 5 of algorithm 1. The size of each data file after processing these steps is shown in Figure 4.

It is observed that all the files contain a lot of redundant data which is not useful for training our model thus has been removed. Every file except DrDDoS NTP has more than 50 percent redundant data which is removed. The shape of each file after processing algorithm 1 is given in figure 5.

The step 5 of Algorithm 1 concatenates all the pre-processed files and again duplicates are checked and if found they are removed from the dataset. Out of 5,00,63,112 samples we have now 82,55,861 samples with a total of 80 features from all the 11 files. The dataset is now processed according to the Algorithm 2.

- *Drop infinity values*

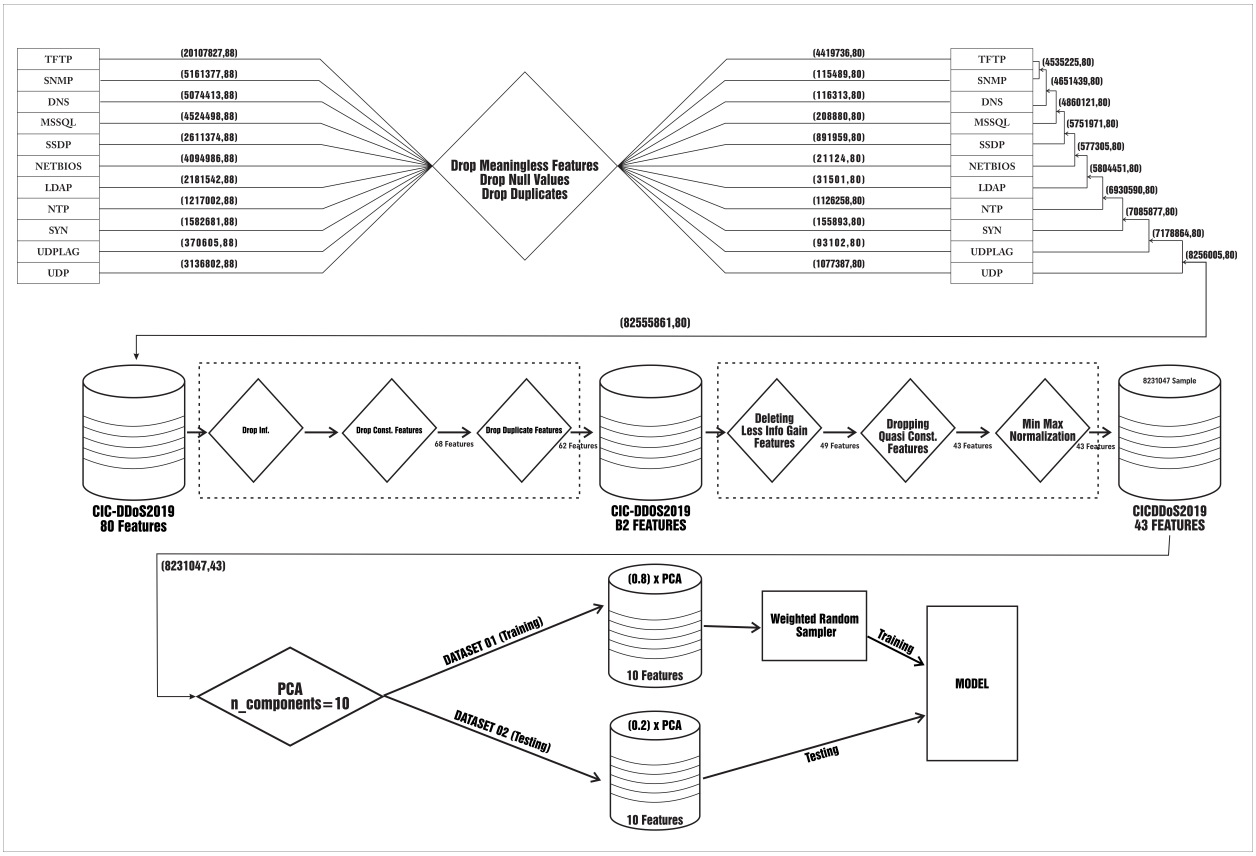

Figure 4: Stages of Pre-processing

Algorithm 2 Data Cleaning 2

Step 1: dataset := drop infinity()
Step 2: $\forall\ F_i \in F$,{ i = 1, 2, 3, ..., N_1}
    if F_i is Constant *then*
        delete F_i in F
Step 3: $\forall\ F_i \in F, \{i = 1, 2, 3, ..., N_2, N_2 < N_1\}$
    if F_i is a duplicate Feature *then*
        delete F_i in F
Step 4: Scale the dataset values between 0 and 1
Step 5: $\forall\ F_i \in F, \{i = 1, 2, 3, ..., N_3, N_3 < N_2\}$
    if inf gain $(F_i) < 0.05$ *then*
        delete F_i in F
Step 6: $\forall\ F_i \in$F, $\{i = 1, 2, ..., N_4, N_4 < N_3\}$
    if Var$(F_i) < 0.01$ *then*
        delete F_i in F
Step 7: dataset = PCAtransformed()
Step 8: Split dataset : train and test; testsize (0.2)

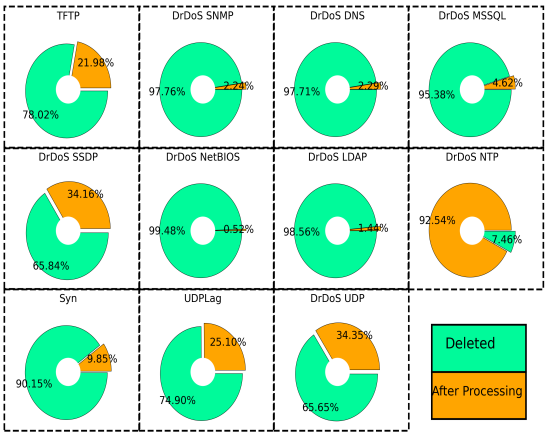

Figure 5: Amount of data before and after processing

Step 1 of algorithm 2 drops infinity values in the dataset. The records containing –inf and inf values are deleted. These values must be imputed for our ML or DL models to work, as discussed earlier if the data at missing is missing not at random (MNAR), the list-wise deletion gives same results as multiple imputation methods so they are dropped from our dataset, and

we are left with 82,55,861 records with 80 features.

- *Drop constant features*

Step 2 of algorithm 2 drops constant features in the dataset. There are two ways to check whether a feature is constant one if the variance of the feature is 0 or if the unique value in the feature is just 1. The algorithm for constant feature elimination is given below.

Algorithm 3 Drop constant features

Step 1: $F_c =: \phi$; F = dataset.columns
Step 2: $\forall\ F_i \in F; \{i = 0, 1, ..., N\}$
if dataset $[F_i]$.unique == 1 *then*
$F_c = F_c \bigcup \mathrm{F}_i$
Step 3 : *Drop* F_c from dataset

Algorithm 3 found there are 12 features having constant values which are *Bwd PSH Flags, Fwd URG Flags, Bwd URG Flags, FIN Flag Count, PSH Flag Count, ECE Flag Count, Fwd Avg Bytes/Bulk, Fwd Avg Packets/Bulk, Fwd Avg Bulk Rate, Bwd Avg Bytes/Bulk, Bwd Avg Packets/Bulk, Bwd Avg Bulk Rate*. All these features are removed from the dataset, and the dataset is left with 68 features.

- *Drop Duplicate Features*

Step 3 of algorithm 2 drops the duplicate features in the dataset. Algorithm for dropping the dropping duplicate features is given below.

Algorithm 4 Drop Duplicate Features

Step 1: $\mathrm{F}_d =: \phi$
Step 2: $F = dataset.columns$
Step 3 : $\forall\ (F_i, F_j) \in F\ \{i \neq j; i, j = \{0, 1, 2, ..., N\}\}$
x =: dataset$[\mathrm{F}_i]$ == $dataset[F_j]$
if $\sum_{i=0}^{M} x_i = \mathrm{M}$ *then*
$F_d = F_d \bigcup \mathrm{F}_j$
Step 4: Drop F_d from the dataset

After analysing there are a total of 6 duplicate features listed as *Subflow Fwd Packets, Subflow Bwd Packets, Subflow Fwd Bytes, Subflow Bwd Bytes, RST Flag Count, Fwd Header Length.1*. The step checks for if any two features represent the same information. These features are also dropped from the dataset, so we are left with 62 features.

- *Scaling the Dataset*

Step 4 scales the whole dataset except the label. The features are scaled using minmax scaler which scales the values between 0 and 1. The formula is given below.

$$X^{scaled} = \frac{X - X_{min}}{X_{max-}X_{min}} \tag{1}$$

- *Drop Less Information Gain Features*

Step 5 deletes the features that have less information gain. The information gain of the feature is calculated using info_gain library and the threshold value is kept 0.05. The algorithm 5 for deletion of less information gain features is given below:

Algorithm 5 Drop Less Information Gain Features

Step 1: $F_{less} =: \phi$
Step 2 : $\forall\ F_i \in \mathrm{F}\ \{i = 0, 1, 2, ..., N\}$
if $\mathrm{info}_g ain\ (\mathrm{F}_i) < 0.05$ *then*
$F_{less} =: F_{less} \bigcup F_i$
Step 3: Drop F_{less} from dataset

In total 13 features were found with information gain less than 0.05 which are *Fwd PSH Flags, Bwd Packet Length Std, Bwd Packet Length Mean, URG Flag Count, Bwd Packet Length Min, SYN Flag Count, Inbound, Bwd Packet Length Max, Avg Bwd Segment Size, Bwd IAT Std, Down/Up Ratio, CWE Flag Count, Total Length of Bwd Packets*. All these features are dropped from the dataset, and we are left with 49 features.

- *Dropping Qasi constant features*

Step 6 of algorithm 2 deletes the qasi constant features in the dataset. These features do not contain much information so dropping these will not impact our model training but will impact the computational cost of the model. Variance threshold is set to less than 0.01. The algorithm is given below:

Algorithm 6 Dropping Qasi constant features

Step 1: $F_{qasi} =: \phi$
Step 2 : $\forall\ F_i \in Fi = 0, 1, 2, ..., N$
var = Var(F_i)
if info gain $(F_i) < 0.05$ then
$F_{qasi} =: \mathrm{F}_{qasi} \bigcup F_i$
Step 3: Drop F_{qasi} from dataset

There are 6 such features which are *Protocol, Flow Duration, Fwd IAT Total, Min Packet Length, ACK Flag Count, and min_seg_size_forward*. All these features are deleted and finally we are left with the 43 features out of which 1 is label.

Now the final pre-processed dataset has a total of 82,55,861 samples with 42 features and 1 label. Then we plotted the pie chart to see the attack types and binary class distributions of the whole pre-processed dataset which is shown in Figure 6 below.

Figure 6 shows the two sets of distributions one for attack types and one for binary. We can see in both the cases we have a huge class imbalance. In attack types 80% of the records are from just three types TTFP, DrDDoS NTP and WebDDoS and 10% from DrDDoS SSDP and rest 10% belong to 9 classes. In binary class distributions 99.39% of the records belong to attack category with just 0.61% belonging to normal category.

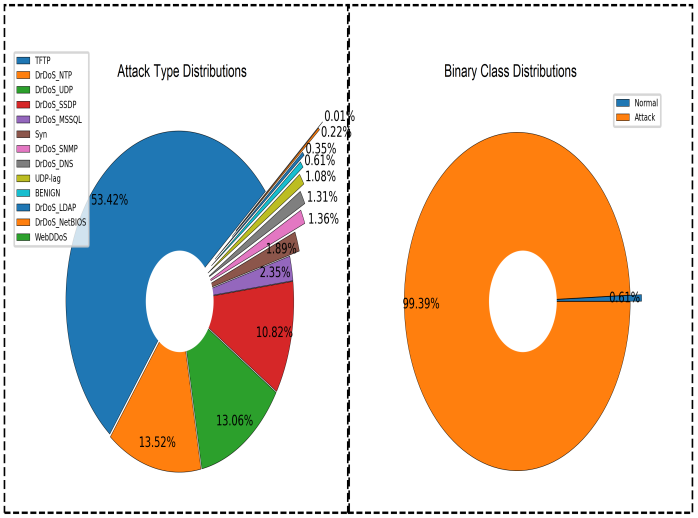

Figure 6: Class distributions of the Final dataset.

- *Feature Reduction*

Principal Component Analysis (PCA) is one of the best feature reduction techniques present in the literature. It is used to convert high dimensional feature space into orthogonal components while retaining the trends and patterns. The components capture variance in the data with highest capture in first component, then second and so on. PCA is used to transform the dataset so that 100% of its variance is captured in fewer components. Instead of using whole feature space, only the minimum components of the transformed data are given to the model for training and testing purpose, thus, contributing to less computational cost of the proposed model. PCA is done on the processed dataset and the first 10 components with their variance capture are plotted in the figure 7 below.

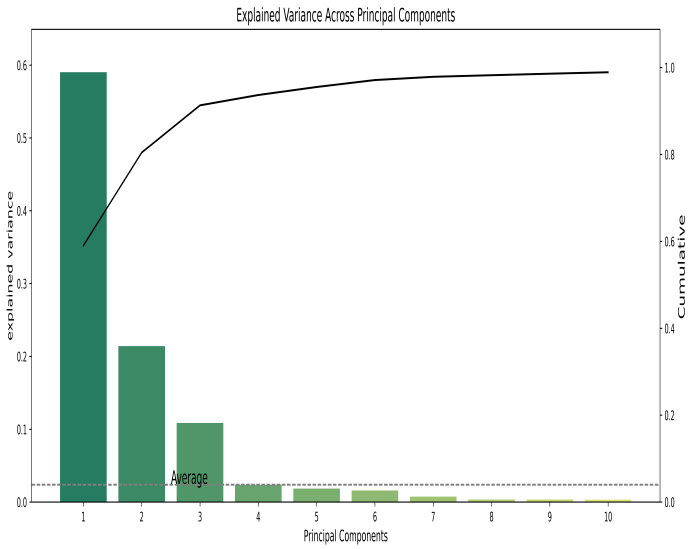

Figure 7: Principal components and their variances.

The figure shows the variance of individual principal components, and it is observed that 10 components capture approximately 100% of the variance in the dataset. So, the final dataset is transformed using the PCA and only 10 components are retained to train our model. Then whole dataset is dividend into two datasets one for training and other for testing with test set size of 20%

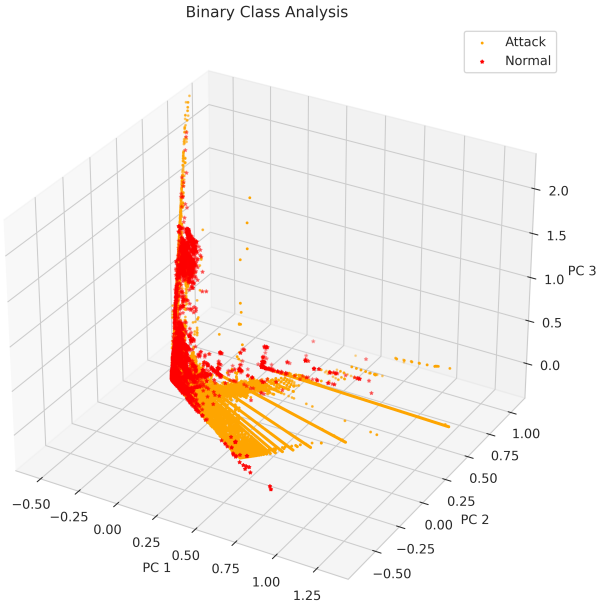

Figure 8: PCA plots of binary class distribution

Figure 8 shows the position of the attack and normal packets with respect to 3 principal components. It is observed that attack and normal packets form a well clusters where attack packets are more in the upward part of the 3d plane and the normal packets are towards the lower part of the 3d plane.

3.2. *Proposed Model Architecture*

The proposed model is built to keep in mind the accuracy and computation cost for predicting whether a pack is malicious or not. The time taken by the intrusion detection system is of utmost importance. If the IDS is taking much time, then they futile and cannot be used in the real world. Figure 9 describes the architecture of our proposed model.

The model has one Conv1D layer with 16 kernels of size 1 and one linear layers having 64 neurons and an output layer with two neurons representing the number of classes. MaxPool2D is used after the CNN layer to reduce the size and dropout of 0.6 is used after the linear layer. Rectified Liner Unit activation function is used to capture the non linearity in the data. The model has just 3298 parameters to be trained thus less computational cost for training as well as testing, resulting into faster detection rate of our IDS.

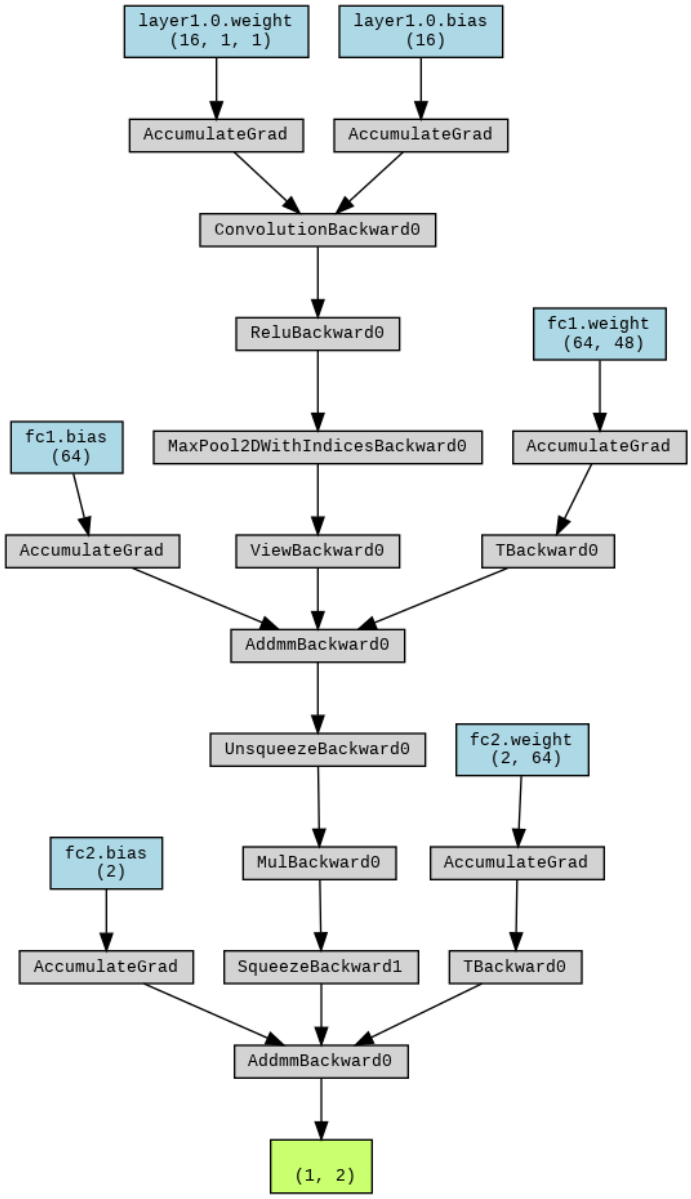

Figure 9: Model Architecture.

4. Experimental Analysis And Results

The section is dedicated to the analysis and evaluation of our proposed model as well as some state of art models proposed by various researchers. The models were implemented and various performance metrics were evaluated.

4.1. Experimental Environment

The experiments were run on a Linux based 20.04 LTS system with intel 3.5GHz processor. The CPU has 16 cores, 128 GB of RAM and GPU from NVIDIA GeForce RTX 2080 with 12 GB of GPU RAM. The link to the repository is https://github.com/owaismujtaba/DDoS-Detection.

4.2. Results

Our model was trained on the train PCA transformed dataset as discussed in the pre-processing section and is evaluated on the test dataset. The BATCH SIZE for the model training is kept 1024, learning rate as 0.01 and Adam optimizer is used for the optimization process. The BinaryCrossEntropy loss function is used in our model.

The proposed model is trained for 100 Epochs. As there are more than 5 million training samples in the dataset each epoch takes 5386 steps to complete. For the validation the test dataset is split into two dataset one as validation and one as test dataset. The validation dataset has around 2.7 million samples and takes 266 steps for each epoch. The model is tested on the remaining test dataset which has around 2.2 million samples. Each epoch takes about 01:02 minutes to train and validation about 0:02 minutes.

The training and validation are shown in the figure 10 below.

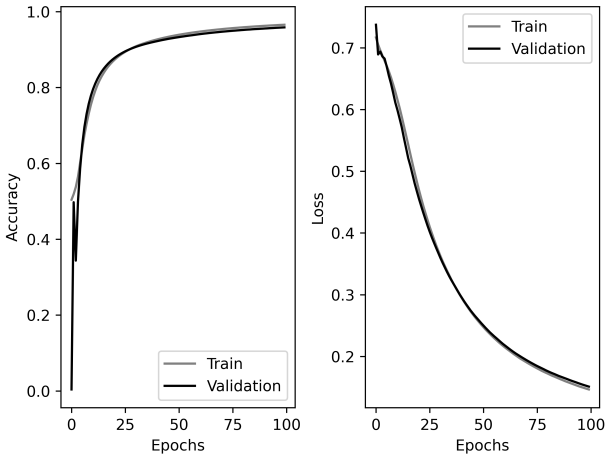

Figure 10: Training and Validation of Proposed Model.

The plot shows the training, validation accuracy and training, validation losses of our proposed model. We can see that the model is learning very well and is increasing drastically from 5% to 80% in 10 epochs and then starts to improve minutely and reaches 99.22% in 100 epochs. The model's performance is validated using validation and can be visualized that the model has learned well and has an accuracy of approximately 100% in validation dataset set as well. The validation accuracy increases from 40% to 99.05% in 100 Epochs. This verifies that the model is not underfitting or overfitting but is able to capture the generalization of the dataset. The training and the validation loss plot shows the decrease of loss from epoch 1 to epoch 100. After 100 epochs the loss on training dataset is 0.0253 and loss on validation dataset is 0.0321. The model is tested on the 20% retained from the full dataset and the models perform with an accuracy of 99.22% , precision of 99.99%, recall of 99.22% and f1 score of 99.60%

Table 3: Comparison of Proposed model to the state of the art results

Author	**Model**	**Year**	**Binary Classification**			
			Accuracy	**Precision**	**Recall**	**F_Score**
DeAssis *et al* [35]	CNN	2020	95.4	93.3	92.4	92.8
Jia *et al* [34]	LSTM	2020	98.9	NA	NA	NA
Ferrag *et al* [24]	CNN	2021	99.95	99	NA	NA
Cil *et al* [5]	CNN	2021	99.97	99.99	99.98	99.98
Raj kumar *et al* [13]	GB	2021	99.97	98.9	99.99	99.44
Devrim *et al* [20]	EFC	2021	99.99	96.15	NA	97.30
yuanyuan wei *et al* [23]	AE-MLP	2021	98.34	97.91	98.48	98.18
Marcos *et al* [22]	GRU	2020	99	99.4	94.7	97
Proposed Method (Ashfaq et al)	**CNN**	**2022**	**99.22**	**99.99**	**99.22**	**99.60**

4.3. Comparison with State of art Studies

Different architectures given by Devrim et al [20] , Raj Kumar et al [13] and Abdullah et al [5] were implemented and tested on the full dataset to check their model's performance using four metrics accuracy, precision, recall and f1 score. The models computational cost is calculated as well in terms of time taken to predict the sample packet to be benign or malicious and is compared with our proposed model.

The proposed model is compared with the different models that have been suggested by the authors done in the literature review and the results are reported in table 3.

Table 4 shows our models performance metrics in comparison with other models. Our model performance is better than most of the models given in the literature and competes with the other models in terms or accuracy and precision. Precision really matters in our case as we cannot afford a malicious packet to be classified as normal. We tested the model on more than 2 million records and have an accuracy of 99.99% which is highest and matches with Cil et.al.

We implemented [13] paper and considered the full dataset that has 11 files and trained their model. The pre-processing suggested by the authors has ambiguity. After EDA they are performing SMOTE for class imbalance, but the dataset has categorical values which must be converted to numerical values, null values and infinity values which must be removed or imputed for the SMOTE to work. Thus, the step data cleaning and imputation should be before SMOTE contradicting the authors. Categorical values were label encoded, infinity and null values were removed to make the model work. The model is implemented and trained on dataset that is pre-processed as suggested by the authors and at the end SMOTE is used to remove the class imbalance problem. The results are reported in the table 4. Our models accuracy is 0.55% less than their model but our model has more precision than reported by the authors.

Another paper [20] is also implemented and the full dataset having 11 files same as previous one. There are infinity values in the dataset which has not be addressed by the authors. The dataset contains categorical variables which are not addressed how to be handled. The infinity and null values were replaced by the median for testing their model. The 8 meaningless features, constant features were removed, and less info gain features were removed. Then their model is trained and tested, the results are reported in table 4. Our model has 0.77% less accuracy than their model but our models Precision is 1 % higher as reported by the authors which matters in our case.

Another paper [5] is implemented and the model proposed is trained and tested on the full dataset. Infinity values are replaced by -1 and null records are deleted from the dataset. The model is trained on the 69 features suggested by the authors and the performance is reported in table 4. The model performs 0.06% less in accuracy, 0.02% less in precision , 0.05% less in recall and 0.03% in fa score. The model takes 27 s to predict a packet to be malicious or attack packet.

Ferrag *etal* [24] reported accuracy of 99.97 % but they have not reported other metrics such as recall and f1 score of their model. Accuracy is not the true performance measure always we have is look at other metrics as well. Our models precision is more than 0.22 than reported by the authors.

Table 4 shows the methods implemented. [13], [5], [20] are implemented and the performance metrics reported by the authors and obtained in the experiments are presented. The models were trained on the 80 % full dataset and tested on 20% dataset. The experiments showed that the raj kumar et al [13] model performed with the same metrics as reported by the authors. The model proposed by the abdullah et al [5] has shown 0.06 % less accuracy and 0.01 % less recall. The model proposed by devrim et al [20] show the same results as proposed by the authors. The Prediction time shows the time taken by the model to predict whether a packet is normal or malicious and the Parameters shows the number of parameters in the model

Table 4: Prediction Time and Performance Comparison of the Proposed model with State of the art models.

Author	**Model**	**Number of Parameters**	**Prediction Time**	**Reported Metrics**				**Experimental Metrics**			
				A	**P**	**R**	**F_Score**	**A**	**P**	**R**	**F_Score**
Marvi *et al* (2020) [37]	LDAP	NA	300 s	99.98	99.11	98.53	NA	NA	NA	NA	NA
Marvi *et al* (2020) [37]	SYN	NA	600 s	99.98	89.97	98.66	NA	NA	NA	NA	NA
Marvi *et al* (2020) [37]	PM	NA	180 s	99.98	99.56	97.91	NA	NA	NA	NA	NA
T. Aytaç *et al* (2020) [38]	KNN	NA	1040 s	99.9	NA	NA	NA	NA	NA	NA	NA
Raj Kumar *et al* (2021) [13]	GB	NA	9.237 μs	99.97	98.9	99.99	99.44	99.97	98.9	99.99	99.44
Abdullah *et al* (2021) [5]	CNN	13,792	27.01 μs	99.97	99.99	99.98	99.98	99.91	99.97	99.93	99.95
Devrim *et al* (2021) [20]	CNN	763,265	228.1 μs	99.99	96.15	NA	97.3	99.89	99.84	99.95	99.89
Proposed Model	**CNN**	**3,298**	**1.535 µs**	**99.22**	**99.99**	**99.22**	**99.60**	**99.22**	**99.99**	**99.22**	**99.60**

architecture.

The prediction time of our model is 1.535 micro seconds which is the lowest compared with other models. Our model is 6 times faster than raj kumar et al [13]. The number of parameters used by the model is shown as well and our model has minimum number of parameters to be trained. 3298 parameters are trained which are 231 times less than devrim et al [20] , 4 times less than abdullah et al [5]. Thus taking very less time for the training as well as prediction. Furthermore, our model has less than 3.5K parameters making it very fast and having low computational cost. The model can be implemented on light weight devices that do not have much computational powers, making it light weight.

5. Conclusion

This study proposes efficient feature extraction technique and LIDS for the detection of DDoS attacks. LIDS: A Light-weight Intrusion Detection System that detect whether a packet is malicious or normal packet is based on deep learning. The model has 1 CNN layer and two linear layers, a total of 3298 parameters to train. The feature extraction technique explained in Data Cleaning Algorithm 1 and 2 is used and only 10 features are used for training our model. The proposed model has less than 3.5K parameters hence, making it a light weight model that can be trained on devices having less computational power. Experimental results show that our model is very fast and accurate. The model predicts a packet is normal or attack packet in less than 1.6 micro seconds fastest among all the models presented in the literature. Our model is 6x, 17x and 148x faster than Rajkumar et al [13], Abdullah et al [5] and Devirim et al [20] respectively. The proposed model predicts with an accuracy of 99.22 % , recall of 99.22 %, precision of 99.99 % and f1 score of 99.60 % .

References

[1] Mittal, M., Kumar, K., Behal, S. (2022). Deep learning approaches for detecting DDoS attacks: a systematic review. Soft Computing, 2(5),(10-29). https://doi.org/10.1007/s00500-021-06608-1.

[2] Mujtaba Khandy, O., Dadvandipour, S. (2021). Analysis of machine learning algorithms for character recognition: a case study on handwritten digit recognition. Indonesian Journal of Electrical Engineering and Computer Science, 21(1), 574. https://doi.org/10.11591/ijeecs.v21.i1.pp574-581

[3] Khanday, O. M., Dadvandipour, S. (2020). Convolutional Neural Networks and Impact of Filter Sizes on Image Classification. Multidiszciplináris Tudományok, 10(1), 55–60. https://doi.org/10.35925/j.multi.2020.1.7

[4] Ahmad, R., Alsmadi, I., Alhamdani, W., Tawalbeh, L. (2022). A Deep Learning Ensemble Approach to Detecting Unknown Network Attacks. Journal of Information Security and Applications, 67, 103196. https://doi.org/10.1016/j.jisa.2022.103196.

[5] Cil, A. E., Yildiz, K., Buldu, A. (2021). Detection of DDoS attacks with feed forward based deep neural network model. Expert Systems with Applications, 169, 114520. https://doi.org/10.1016/j.eswa.2020.114520.

[6] Cisco (2020). *Cisco Annual Internet Report (2018–2023)* White Paper. Retrieved from https://www.cisco.com. Accessed 05 May 2022.

[7] Alethea Toh (2022). *Azure DDoS Protection—2021 Q3 and Q4 DDoS attack trends*. Retrieved from https://azure.microsoft.com/en-us/blog/azure-ddos-protection-2021-q3-and-q4-ddos-attack-trends/. Accessed 05 May 2022.

[8] Ismail, Mohmand, M. I., Hussain, H., Khan, A. A., Ullah, U., Zakarya, M., Ahmed, A., Raza, M., Rahman, I. U., Haleem, M. (2022). A Machine Learning-Based Classification and Prediction Technique for DDoS Attacks. IEEE Access, 10, 21443–21454. https://doi.org/10.1109/access.2022.3152577.

[9] Najar, A. A., Manohar Naik, S. (2022). DDoS attack detection using MLP and Random Forest Algorithms. International Journal of Information Technology, 14(5), 2317–2327. https://doi.org/10.1007/s41870-022-01003-x.

[10] Gaurav, A., Gupta, B. B., Panigrahi, P. K. (2022). A novel approach for DDoS attacks detection in COVID-19 scenario for small entrepreneurs. Technological Forecasting and Social Change, 177, 121554. https://doi.org/10.1016/j.techfore.2022.121554.

[11] Batchu, R. K., Seetha, H. (2022). On improving the performance of DDoS attack detection system. Microprocessors and Microsystems, 93, 104571. https://doi.org/10.1016/j.micpro.2022.104571.

[12] Nuiaa, R. R., Manickam, S., Alsaeedi, A. H., Alomari, E. S. (2022). A new proactive feature selection model based on the enhanced optimization algorithms to detect DRDoS attacks. International

Journal of Electrical and Computer Engineering (IJECE), 12(2), 1869. https://doi.org/10.11591/ijece.v12i2.pp1869-1880.

[13] Batchu, R. K., Seetha, H. (2021). A generalized machine learning model for DDoS attacks detection using hybrid feature selection and hyperparameter tuning. Computer Networks, 200, 108498. https://doi.org/10.1016/j.comnet.2021.108498.

[14] Khoei, T. T., Aissou, G., Hu, W. C., Kaabouch, N. (2021, May 1). Ensemble Learning Methods for Anomaly Intrusion Detection System in Smart Grid. IEEE Xplore. https://doi.org/10.1109/EIT51626.2021.9491891

[15] Alamri, H. A., Thayananthan, V. (2020). Bandwidth Control Mechanism and Extreme Gradient Boosting Algorithm for Protecting Software-Defined Networks Against DDoS Attacks. IEEE Access, 8, 194269–194288. https://doi.org/10.1109/access.2020.3033942.

[16] Pontes, C. F. T., de Souza, M. M. C., Gondim, J. J. C., Bishop, M., Marotta, M. A. (2021). A New Method for Flow-Based Network Intrusion Detection Using the Inverse Potts Model. IEEE Transactions on Network and Service Management, 18(2), 1125–1136. https://doi.org/10.1109/tnsm.2021.3075503.

[17] Varghese, J. E., Muniyal, B. (2021). An Efficient IDS Framework for DDoS Attacks in SDN Environment. IEEE Access, 1–1. https://doi.org/10.1109/access.2021.3078065.

[18] Maseer, Z. K., Yusof, R., Bahaman, N., Mostafa, S. A., Foozy, C. F. M. (2021). Benchmarking of Machine Learning for Anomaly Based Intrusion Detection Systems in the CICIDS2017 Dataset. IEEE Access, 9, 22351–22370. https://doi.org/10.1109/access.2021.3056614.

[19] Gohil, M., Kumar, S. (2020, December 1). Evaluation of Classification algorithms for Distributed Denial of Service Attack Detection. IEEE Xplore. https://doi.org/10.1109/AIKE48582.2020.00028.

[20] AKGUN, D., HIZAL, S., CAVUSOGLU, U. (2022). A New DDoS Attacks Intrusion Detection Model Based on Deep Learning for Cybersecurity. Computers Security, 102748. https://doi.org/10.1016/j.cose.2022.102748.

[21] G.C. Amaizu, C.I. Nwakanma, S. Bhardwaj, J.M. Lee, D.S. Kim (2021). Composite and efficient DDoS attack detection framework for B5G networks, *Computer Networks*, Volume 188, 107871, ISSN 1389-1286, https://doi.org/10.1016/j.comnet.2021.107871.

[22] Assis, M. V. O., Carvalho, L. F., Lloret, J., Proença, M. L. (2021). A GRU deep learning system against attacks in software defined networks. Journal of Network and Computer Applications, 177, 102942. https://doi.org/10.1016/j.jnca.2020.102942.

[23] Wei, Y., Jang-Jaccard, J., Sabrina, F., Singh, A., Xu, W., Camtepe, S. (2021). AE-MLP: A Hybrid Deep Learning Approach for DDoS Detection and Classification. IEEE Access, 9, 146810–146821. https://doi.org/10.1109/access.2021.3123791.

[24] Ferrag, M. A., Shu, L., Djallel, H., Choo, K.-K. R. (2021). Deep Learning-Based Intrusion Detection for Distributed Denial of Service Attack in Agriculture 4.0. Electronics, 10(11), 1257. https://doi.org/10.3390/electronics10111257.

[25] Shieh, C.-S., Lin, W.-W., Nguyen, T.-T., Chen, C.-H., Horng, M.-F., Miu, D. (2021). Detection of Unknown DDoS Attacks with Deep Learning and Gaussian Mixture Model. Applied Sciences, 11(11), 5213. https://doi.org/10.3390/app11115213.

[26] Sanchez, O. R., Repetto, M., Carrega, A., Bolla, R. (2021, June 1). Evaluating ML-based DDoS Detection with Grid Search Hyperparameter Optimization. IEEE Xplore. https://doi.org/10.1109/NetSoft51509.2021.9492633.

[27] Singh Samom, P., Taggu, A. (2021). Distributed Denial of Service (DDoS) Attacks Detection: A Machine Learning Approach. Applied Soft Computing and Communication Networks, 75–87. https://doi.org/10.1007/978-981-33-6173-7_6.

[28] Can, D.-C., Le, H.-Q., Ha, Q.-T. (2021). Detection of Distributed Denial of Service Attacks Using Automatic Feature Selection with Enhancement for Imbalance Dataset. Intelligent Information and Database Systems, 386–398. https://doi.org/10.1007/978-3-030-73280-6_31

[29] Elsayed, M. S., Le-Khac, N.-A. ., Dev, S., Jurcut, A. D. (2020, August 1). DDoSNet: A Deep-Learning Model for Detecting Network Attacks. IEEE Xplore. https://doi.org/10.1109/WoWMoM49955.2020.00072.

[30] Almiani, M., AbuGhazleh, A., Jararweh, Y., Razaque, A. (2021). DDoS detection in 5G-enabled IoT networks using deep Kalman backpropagation neural network. International Journal of Machine Learning and Cybernetics. https://doi.org/10.1007/s13042-021-01323-7.

[31] Hussain, F., Abbas, S. G., Husnain, M., Fayyaz, U. U., Shahzad, F., Shah, G. A. (2020). IoT DoS and DDoS Attack Detection using ResNet. 2020 IEEE 23rd International Multitopic Conference (INMIC). https://doi.org/10.1109/inmic50486.2020.9318216.

[32] Asad, M., Asim, M., Javed, T., Beg, M. O., Mujtaba, H., Abbas, S. (2019). DeepDetect: Detection of Distributed Denial of Service Attacks Using Deep Learning. The Computer Journal. https://doi.org/10.1093/comjnl/bxz064.

[33] Shurman, M., Khrais, R., Yateem, A. (2020). DoS and DDoS Attack Detection Using Deep Learning and IDS. The International Arab Journal of Information Technology, 17(4A), 655–661. https://doi.org/10.34028/iajit/17/4a/10.

[34] Jia, Y., Zhong, F., Alrawais, A., Gong, B., Cheng, X. (2020). FlowGuard: An Intelligent Edge Defense Mechanism Against IoT DDoS Attacks. IEEE Internet of Things Journal, 7(10), 9552–9562. https://doi.org/10.1109/jiot.2020.2993782.

[35] de Assis, M. V. O., Carvalho, L. F., Rodrigues, J. J. P. C., Lloret, J., Proença Jr, M. L. (2020). Near real-time security system applied to SDN environments in IoT networks using convolutional neural network. Computers Electrical Engineering, 86, 106738. https://doi.org/10.1016/j.compeleceng.2020.106738.

[36] [CICDDoS2019] Canadian Institute for Cybersecurity (2019) https://www.unb.ca/cic/datasets/ddos-2019.html.

[37] Marvi, M., Arfeen, A., Uddin, R. (2020). A generalized machine learning-based model for the detection of DDoS attacks. International Journal of Network Management, 31(6). https://doi.org/10.1002/nem.2152.

[38] Aytac, T., Aydin, M. A., Zaim, A. H. (2020). Detection DDOS Attacks Using Machine Learning Methods. Electrica, 20(2), 159–167. https://doi.org/10.5152/electrica.2020.20049.