OPTIMIZED INTRUSION DETECTION SYSTEM WITH FEATURE EXTRACTION FOR EFFECTIVE NETWORK TRAFFIC CLASSIFICATION

Report of Internship

TATA CONSULTANCY SERVICES (TCS)

15-Nov-2021 to 27-May-2022

Submitted by

Ms. SARANYA T

Intern Emp ID: 2170618

Final-year Postgraduate student

Centre for Artificial Intelligence

THANGAL KUNJU MUSALIAR COLLEGE OF ENGINEERING KERALA

Under the guidance of

RAJEEV AZHUVATH (TCS Mentor ID: 120914)

&

Prof. SUMOD SUNDAR (TKMCE)

MAY 2022

ACKNOWLEDGEMENT

A successful project is a fruitful culmination of efforts by many people, some directly involved and some others indirectly, by providing support and encouragement. Firstly I would like to thank the almighty for giving me the wisdom and grace to make my project a successful one. I thank him for steering me to the shore of fulfilment under his protective wings.

With a profound sense of gratitude, I would like to express my heartfelt thanks to my mentor **Rajeev Azhuvath, Tata Consultancy Services (TCS)**, for his expert guidance, constant support and cooperation.

I express my sincere gratitude to **Dr. T A Shahul Hameed**, Principal of TKMCE and **Dr. Imthias Ahamed**, Professor and Head of the Department, Centre for Artificial Intelligence, TKMCE, for their immense encouragement. I would like to thank my college guide **Prof. Sumod Sundar**, Assistant Professor, Centre for Artificial Intelligence, TKMCE, for his expert guidance and cooperation.

I also express my thanks to my loving parents, brother and friends, for their support and encouragement in the successful completion of this project work.

SARANYA T

Abstract

With the advancement in internet technology, it is now possible to collect and transfer data over multiple networks easily. An increase in network-related threats has led to several privacy concerns. Intrusion Detection System (IDS) can detect malicious network patterns those conventional systems like firewalls cannot easily detect. There are different types of IDS, but those systems are prone to high False Alarm Rate (FAR) because anomalies can be new legitimate activities. Good quality and surplus network traffic patterns will make IDS systems more effective. Hence, to reduce the FAR and enhance detection accuracy, we propose a novel method to extract the best features and classify them as 'normal' and 'intrusive'. Temporal relations in the dataset are also analyzed. Feature extraction is done with XGBoost and Autoencoder in this work, and classification is done with TabNet. The proposed method is compared with four classifier models - XGBoost, Dense Neural Network, Convolutional Neural Network, and Temporal Convolution Network. Results prove that XGBoost feature extraction is a more efficient method for reliable feature extraction when compared to Autoencoder. The proposed model, XGBoost feature extraction, and TabNet-based classification provided maximum detection accuracy in UNSW-NB15 and NSL-KDD datasets.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With an increase in accessibility of the internet, it became possible to fetch and transfer data and information over multiple networks easily. But the vital task is to protect such systems and networks from data disclosure. This led to the development of automated techniques in cybersecurity domain. But the spike in network-related threats such as worms, malware, viruses, etc., demands a system that can adequately monitor the network activity to detect threats that interrupt the security systems.

IDS is one such model that properly monitors and studies the pattern of a network traffic which is vulnerable to malicious activity and makes the operators vigilant by alarms or alerts. This system works on the principle that the conduct of a typical user and an intruder is different, which makes it possible to distinguish between normal and malicious activities. Typically, there are three types of IDS system. This model cannot detect new intrusions, resulting in a high FAR. High FAR is a challenge in these systems because anomalies can be legitimate activities. Another type of IDS was developed with the advancements in machine learning techniques. For accurate detection of intrusions in such models, it is necessary to have good quality and quantity of traffic data. So, one of the crucial tasks of researchers while developing IDS is to extract relevant features from such large quantities of data. To identify the best features and improve the detection accuracy, researchers proposed several approaches involving machine learning techniques, deep learning techniques, and hybrid techniques.

This work focuses on developing an efficient IDS system that can reduce FAR and improve the detection rate in terms of accuracy. Hence, we propose a novel method with XGBoost feature extraction and TabNet-based classification to identify the best features and classify them as 'normal' and 'intrusive'. The contribution of this work includes the following: -

- Two standard datasets: - UNSW-NB15 and NSL-KDD, were used for experimentation with the proposed system.
- Select the best encoding technique.
- XGBoost and Autoencoder-based feature extraction.
- Implementing TabNet classifier model for classifying intrusive and normal traffic.
- Comparison of the proposed model with XGBoost, Dense Neural Network (DNN), Convolutional Neural Network (CNN), and Temporal Convolutional Network (TCN) classifiers.

Chapter 2

Related Works

In this section, several studies of intelligent network IDS utilizing both machine learning and deep learning techniques are discussed.

Devan et al. [1] developed a feature selection model with XGBoost (eXtreme Gradient Boosting) followed by a DNN based classifier. The main problem addressed is it's high FAR. Simple machine learning models require superior domain knowledge for identifying relevant patterns. With deep learning techniques, faster detection of attacks is possible. Hence a model combining both machine learning and deep learning techniques is proposed. NSL-KDD dataset is used for experimentation in this work. More in-depth study of attack classes is yet to be analyzed.

Kasongo et al. [2] proposed a Deep Long Short-Term Memory (DLSTM) based classifier incorporating information gain-based feature scoring. The directional loop in Recurrent Neural Network (RNN) memorize the previous state, but the vanishing gradient issue reduce the accuracy in such models. Forget gate in LSTM, which can omit the irrelevant data, is utilized to develop the model. The NSL-KDD dataset is used to train and evaluate the model. But the work generally focused on the malicious class rather than studying individual classes separately.

Moustakidis et al. [3] developed a deep learning-based Siamese Convolutional Neural Network (SCNN) for extracting the relevant features from surplus network data. This model is proposed to develop a user-friendly risk indicator for identifying cyber-attacks. The NSL-KDD dataset is used to train and evaluate the model. Pre-processing is carried out with a fuzzy allocation scheme whose output is fuzzy values. These fuzzy values are then converted into images with a Vec2im-based modality transformation technique. The output obtained from Vec2im then serves as input to SCNN. The results revealed that the proposed system ensure capability in detecting attacks. But it is difficult to apply the proposed model where decisions should be taken based on complex and heterogenous data.

Alhajjar et al. [4] developed a model to evaluate the sensitivity of various machine learning models when they are undergone adversarial attacks. Adversarial systems are systems that are developed intentionally to fool the original model. So, to test the capacity of simple machine learning classifiers, training is done in both NSL-KDD and UNSW-NB15 datasets. Initially, adversarial examples are generated with Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Generative Adversarial Network (GAN), and Monte Carlo Simulation (MC). Each model produces malicious vectors, which are then input into classification models, including Support Vector Machine (SVM), Naïve Bayes, Decision Tree (DT), Random

Forest (RF), K-Nearest Neighbor (KNN), Multilayer perceptron, Gradient Boosting, Linear regression, Bagging, etc. With the proposed model, SVM and DT are identified as more vulnerable, and hence it is suggested to hold back these models from using it in automatic IDS systems. But it is impossible to analyze why some models are more robust than others as there is no system to identify the internal operations of the machine learning models.

Nguyen et al. [5] proposed an extensive feature selection method with a GA and KNN incorporating a 5-fold cross-validation fitness function along with Fuzzy C-means Clustering to develop an Improved Feature Subset (IFS). 3 CNN models are selected, and the IFS is served as input to these models. Further, in-depth features are extracted from these CNN models to form a Deep Feature Subset (DFS). The training and testing are done in the NSL-KDD dataset. Performance comparison is carried out by applying IFS and Original Feature Subset (OFS) in different classification models. But the system consumed more time to implement the GA algorithm and select the best 3 CNN models in the initial phase.

Rajagopal et al. [6] proposed an ensemble-based classification model for IDS systems incorporating hashing and information gain techniques for feature selection. The detection accuracy of hybrid models is comparatively more than simple machine learning models. RF, KNN, and Logistic regression are used as the base classifiers in this system, and SVM is chosen as the meta-model. Training and testing are performed in the UNSW-NB15 dataset. The proposed model obtained an accuracy of 92.85%. Since the classifier output is based on the maximum voting scheme, the imbalance in the dataset has not become an issue. Even though FAR is substantially reduced the model takes enough time for processing the data.

Asahi-Shahri et al. [7] developed a genetic engineering-based feature selection model with embedded parameter optimization. Finally, the classification is done with SVM. KDD Cup dataset is used to check the effectiveness of the system. Thereby, the FAR is reduced. But the training time and testing time required are more than other models.

Hsu et al. [8] incorporating two deep learning architectures such as simple LSTM model and CNN – LSTM hybrid model. Both binary and multi-class classification is done using instances from the NSL-KDD dataset. Both models performed much better than a simple RNN-based IDS system. The research proved that incorporating CNN before LSTM improved the accuracy as convolutional layers can extract the local features. LSTM time-series data is also studied.

Gao et al. [9] developed a system that can detect both temporally uncorrelated and correlated attacks. Three approaches were introduced for the temporal feature study. These include: - Feed Forward Neural Network (FNN), LSTM, and an architecture combining FNN and LSTM by ensemble approach. Each data packet has different features of network traffic, and in-packet features such as sequencing data, error codes, etc., are not enough to detect time-correlated attacks. So, from a simple FNN model, the accuracy in detecting correlated attacks is less. The accuracy of detection is increased by using LSTM that predict future timestamps when the previous time stamps are known. But the proposed system did not effectively utilize convolutional layers to study the local features.

Chapter 3

Intrusion Detection System

An intrusion detection system (IDS) monitors network traffic patterns and helps distinguish between legitimate and malicious traffic. As its name suggests, this is used to detect different attacks. An IDS system can be deployed in small scale systems or large scale systems, i.e., its scope can range from single computers to large networks of computers. An attack can be either host-based or network-based.

3.1 Challenges of IDS

IDS can either be signature-based or anomaly-based. Signature-based IDS, otherwise known as misuse-based IDS system, is developed so that any pre-defined patterns of network traffic will be easily identified. In contrast, Anomaly-based IDS detects attacks based on variations from the normal instances which are already defined. The main disadvantage of a signature-based system is that it will not be able to identify unseen attacks. But the deployment of both techniques could not reduce the problem of false alarms, i.e., all these techniques are based on strict rules, and they are vulnerable to false positive and false negative alarms. Apart from that, the computing cost is high as the network traffic types increase day by day. The network characteristics are becoming complex as the attackers are changing the features periodically.

3.2 Problem under analysis

3.2.1 Enhancing the detection rate of IDS with optimized FAR

An authentic IDS can be established with a reliable feature selection technique. Several features can be extracted from data packets. Efficient detection of attacks requires identification of relevant features. This can be extracted using machine learning algorithms. But in such cases, there is chance that some intrusion attacks are misclassified as normal, and some normal flows are misclassified as intrusions and thereby increasing the FAR.

Chapter 4

Proposed system architecture and dataset description

4.1 Methodology

The proposed system architecture for effective network traffic classification is shown in fig 3.1. Data preprocessing, parameter tuning, feature extraction, and classification are the four essential tasks for accomplishing this work. The steps include: -

1. Data preprocessing includes encoding, resampling, and normalization.
2. Perform parameter tuning to assign the best values to the model.
3. Find the best correlations and develop the feature score graph using the feature extraction technique.
4. Apply features to the classification model.
5. Compare their performance and evaluate the FAR with performance metrics like accuracy, precision, recall, F1-score, and ROC-AUC score.

4.2 Dataset description

4.2.1 UNSW-NB15 Dataset

The raw network packets of the UNSW-NB15 data set are created by the IXIA Perfect Storm tool in the Cyber Range Lab of the Australian Centre for Cyber Security (ACCS) for generating a hybrid of actual modern normal activities and synthetic recent attack behaviours. The tcpdump tool captures 100 GB of the raw traffic (e.g., pcap files). This dataset contains nine classes of attacks, namely: Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The Argus and Bro-IDS tools are utilized, and twelve algorithms are developed to generate 49 features with the class label. A partition from this data set is configured as a training set and testing set. The training set contains 175,341 records, while the testing set contains 82,332 records from various classes of 'intrusive' and 'normal'.

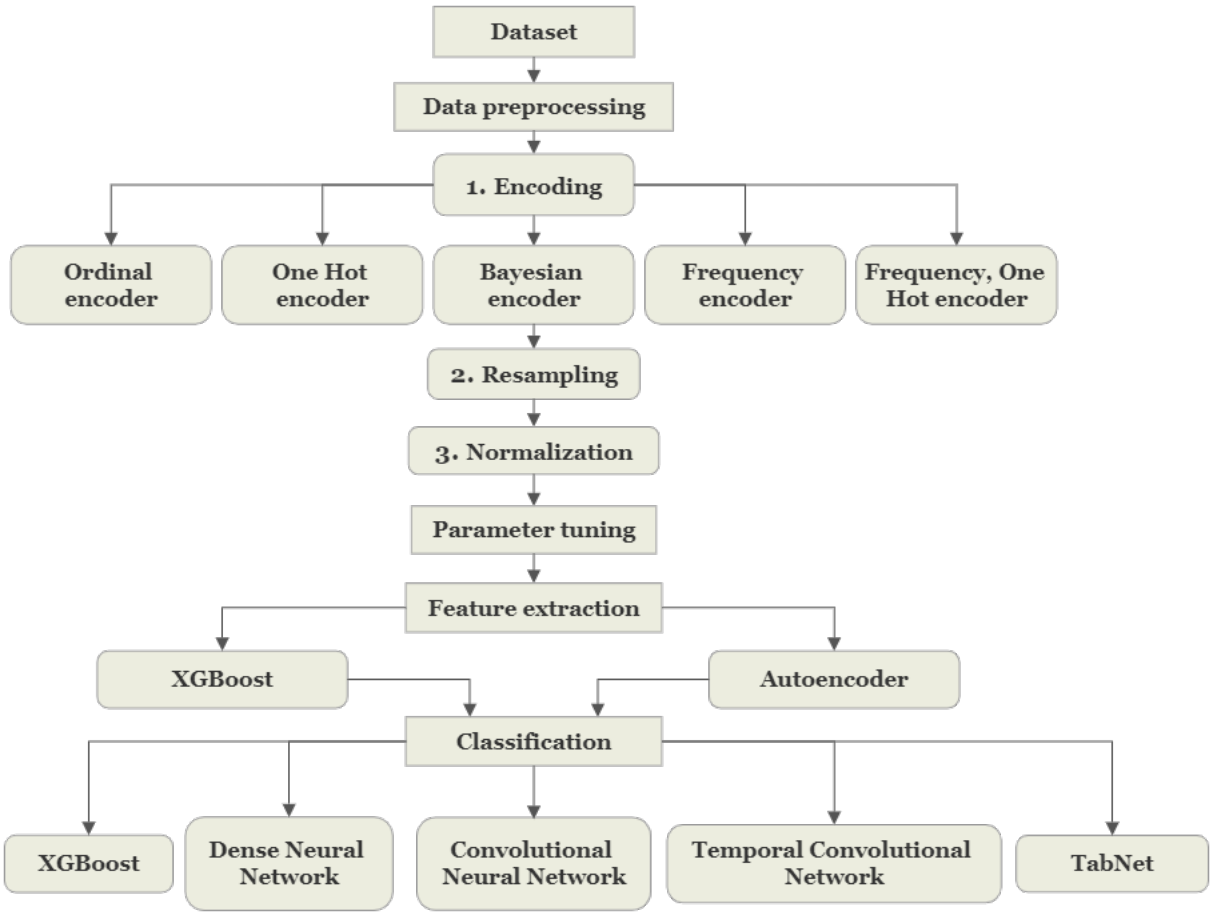

Figure 4.1: Flowchart of proposed model

4.2.2 NSL-KDD Dataset

The NSL-KDD data set is a new version of the KDD'99 data collection. This is a useful benchmark data set for academics to compare various IDS. The training and testing dataset were available in txt format. It is then converted to csv format for our experiments. This dataset contains thirty-nine families of attacks, namely: Neptune, Satan, Upsweep, Smurf, Portsweep, Nmap, Back, Guess_passwd, Mscan, Warezmaster, Teardrop, Warezclient, Apache2, Processtable, Snmpguess, Saint, Mailbomb, Pod, Snmpgetattack, Httptunnel, Buffer_overflow, Multihop, Land, Rootkit, Named, Ps, Sendmail, Xterm, Imap, Ftp_write, Loadmodule, Clock, Phf, Perl, Xsnoop, Spy, Worm, Udpstorm, Sqlattack. All these categories were converted into a single "intrusive" category. The dataset is partitioned into training and testing sets. The training set contains 125,973 records, while the testing set contains 22,544 records from various classes of 'intrusive' and 'normal'.

This is a binary classification problem; hence all the attack categories are labelled into a single category of 'intrusive'. So, now the dataset consists of two labels: - '0' for normal and '1' for intrusion. Redundant columns are discarded manually.

Chapter 5

Experimental setup and results

The hardware used for the experiments includes Windows 10 Pro OS, 64-bit operating system, x64-based processor, Intel(R) Core (TM) i3-5005U CPU @ 2.00GHz, 2.00 GHz, 4 GB RAM. The experimental environment was prepared by using Python 3.7 programming language. The framework used is Keras with TensorFlow as background in the Anaconda environment. Machine learning and deep learning libraries include - NumPy, Pandas, Matplotlib, and Scikit learn. Performance analysis identifies the best model having the highest detection rate. The general evaluation metrics such as Accuracy, Precision, Recall, F1 score, ROC AUC score, and confusion matrix are used.

5.1 Data Preprocessing

5.1.1 Encoding

Encoding is one of the main steps in data preprocessing. Every encoder has its characteristic feature, distinguishing it from every other encoder. So, to identify which characteristic feature suits our application, experiments on encoding are done with five different combinations: - Ordinal encoding, one-hot encoding, frequency encoding, combining one hot and frequency encoding, and Bayesian encoding. Initially, the required libraries and the dataset is loaded. Missing values and count of categories in each categorical feature column are identified. Each of these categorical features is encoded separately with five encoder combinations and is saved in a separate csv file. Without further preprocessing, the five sets of the UNSW-NB15 dataset and five sets of the NSL-KDD dataset are applied to the XGBoost classifier model to identify the best encoder. Table 5.1 shows the performance metric values obtained after encoding. As shown in this table, the Bayesian encoder gives the highest accuracy.

The Bayesian encoder provides good results because of its ability to study how a feature is dependent on the target data. Compared to other classic encoders where the relation between each category within the same feature is considered, target or Bayesian encoding evaluates the mean value of a particular category based on its occurrence with the target. Hence, it is the best among all the combinations in both datasets. Since Bayesian encoding is proved to be robust, Bayesian encoded UNSW-NB15, and Bayesian encoded NSL-KDD dataset is used for further experimentation.

Processing tool	Accuracy(%)	
	UNSW-NB15 dataset	NSL-KDD dataset
Ordinal Encoder	55.06	77.44
One Hot and Frequency encoder	87.59	77.59
One Hot Encoder	87.61	78.21
Frequency encoder	87.72	77.37
Bayesian encoder	**90.12**	**79.79**

Table 5.1: Experimental result of data preprocessing with different encoders

5.1.2 Resampling and Normalization

The next step in data preprocessing is resampling. In UNSW-NB15 dataset, there are 119,341 "attack" instances and 56,000 "normal" instances and in NSL-KDD dataset, there are 67,343 "attack" instances and 58,630 "normal" instances. After performing SMOTE-based resampling, the number of training instances in the UNSW-NB15 and NSL-KDD datasets increased from 175,341 to 238,682 and 125,973 to 134,686, respectively, i.e., the minority class label is now in proportion with the majority class.

The final step in data preprocessing includes normalizing the data. Both UNSW-NB15 and NSL-KDD dataset is normalized with min-max normalizer to improve the detection accuracy. The new range of normalization is fixed as range_min = 0 and range_max= 1.

5.2 Parameter tuning

XGBoost has some intrinsic characteristics, which make it robust and efficient. But parameter tuning is done to identify the best parameters which will offer a sound detection rate corresponding to a particular dataset, here UNSW-NB15 and NSL-KDD dataset. Grid search algorithm is utilized for the same. The n-estimators are successfully identified by setting the initial n-estimate as '4000' and '1000' in UNSW-NB15 and NSL-KDD datasets, respectively, based on the total number of data samples in each dataset. Initially, the model is created by assigning random values. Finally, parameters like 'max_depth', 'min_child_weight', 'gamma', 'subsample', 'col_sample_bytree', and 'alpha' is tuned. Table 5.2 shows the values obtained after tuning parameters for both datasets.

5.3 Feature extraction

5.3.1 Feature extraction with XGBoost

After loading Bayesian encoded training and testing data with pandas, both dataset's feature score graph and f-score value are obtained. The sort () imported from NumPy identified the thresholds in ascending order. Figs 5.1 and 5.2 show the feature importance graph of UNSW-NB15 and NSL-KDD datasets.

Parameter	Value Obtained	
	UNSW-NB15 dataset	NSL-KDD dataset
n_estimators	1484	381
max_depth	4	5
min_child_weight	5	8
gamma	0.4	0.0
subsample	0.6	0.9
col_sample_bytree	0.6	0.6
alpha	0.05	0.005

Table 5.2: Experimental result of XGBoost Parameter tuning

Tables 5.3 and 5.4 show the result of XGBoost feature selection with different thresholds. With the decrease in the number of features, the model's performance decreases. Hence, we need to find the highest accuracy with the optimum number of features selected. The highest accuracy of 90.53% with a threshold = 0.016 and 15 selected features is the best combination in the UNSW-NB15 dataset. Similarly, in the NSL-KDD dataset highest accuracy of 82.10% with a threshold = 0.065 and 4 selected features are the best combination.

Threshold	Number of features	Accuracy (%)
0.000	42	90.20
0.001	40	90.22
0.002	38	90.20
0.003	33	90.39
0.004	26	90.48
0.004	24	90.34
0.004	23	90.37
0.004	20	90.18
0.016	**15**	**90.53**
0.016	11	86.91
0.016	6	81.03
0.016	2	80.57

Table 5.3: XGBoost feature extraction with different thresholds in UNSW-NB15 dataset

5.3.2 Feature extraction with Autoencoder

Imported all the required packages and datasets. Created input, encoder, bottleneck, decoder, and an output layer of the autoencoder. Input layers consist of units as the number of features in the UNSW-NB15 and NSL-KDD dataset. 2 encoder layers, each with a dense layer, batch normalization layer, and leaky ReLU layer. The first layer contains 86 units, and the second layer contains 43 units. The bottleneck layer (dense layer) has 21 units. The decoder part has two layers: a dense layer, a batch normalization layer, and a leaky ReLU layer. Output layer with Linear activation function, optimizer = 'adam', loss='mse'. Trained

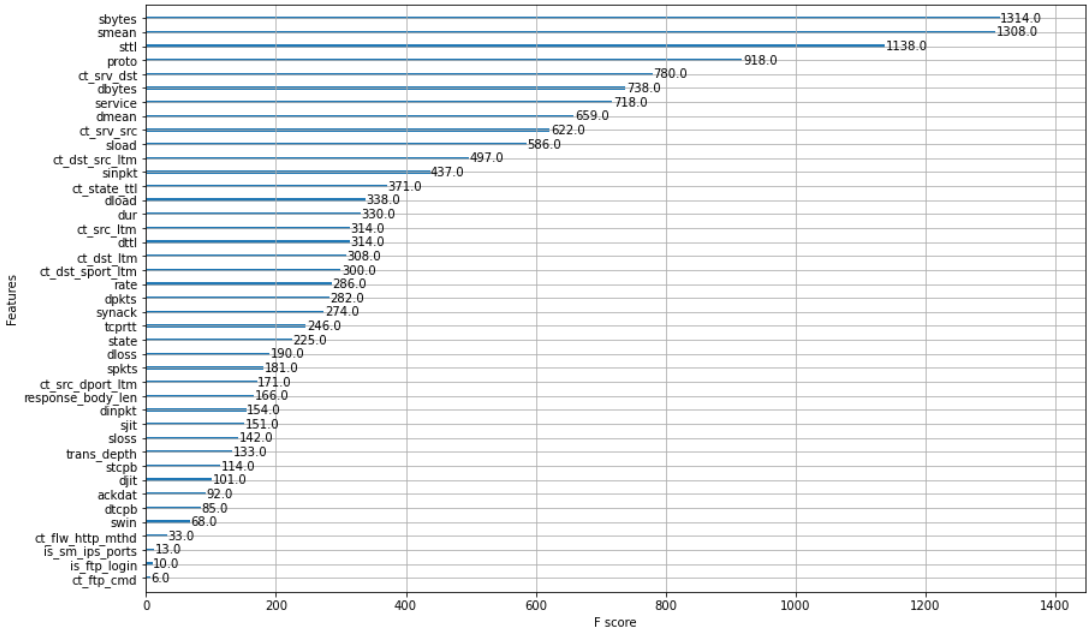

Figure 5.1: Feature importance graph of UNSW-NB15 dataset

the model and then saved the model with output as a bottleneck. Fit the saved autoencoder model data into the XGBoost classifier model and perform classification to identify the best feature extraction algorithm. Table 5.5 shows both dataset's Autoencoder and XGBoost feature selection model results. Accuracy of 85.32% and 75.94% is obtained in UNSW-NB15 and NSL-KDD datasets, respectively.

From table 5.5, the XGBoost feature extraction model provides the highest detection accuracy in both datasets. This is because, for structured tabular data, XGBoost consistently outperforms. Just applying normalization to tabular data and training a model, XGBoost gives better results. XGBoost can minimize the loss of what is learned and add a new tree to it. This sequential learning allows XGBoost to extract features more efficiently. But in the case of unstructured data, neural networks outperform. Hence, we can infer that XGBoost feature selection is more effective than Autoencoder-based feature extraction in intrusion datasets.

5.4 Classification and comparative analysis

The detection rate with the proposed model XGBoost feature selection and TabNet classification technique yields a maximum accuracy of 93.02% and 88.35% compared to all the models in the UNSW-NB15 and NSL-KDD dataset. Structured tabular data and a normalization help XGBoost perform well. This is because of its ability to minimize the loss of what is learned and then add a new tree. Autoencoder requires further optimization to enhance the effectiveness of feature extraction in tabular data.

TabNet is a tabular deep learning model. Each decision step employs sequential attention

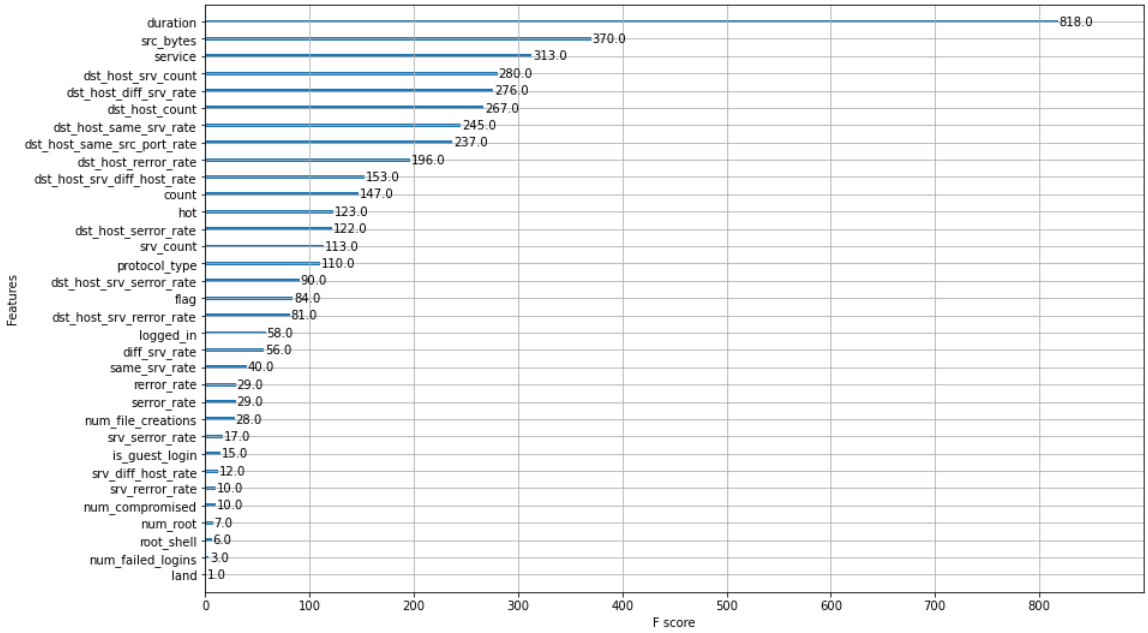

Figure 5.2: Feature importance graph of NSL-KDD dataset

to select a subset of relevant features and process them locally and globally, i.e., for a single input and the whole dataset. TabNet model provides more accuracy when compared to XGBoost, DNN, CNN, and TCN. The 2-stage feature selector in the TabNet model helps in more robust classification. Hence, even though XGBoost provides good feature extraction results, in classification, TabNet outperforms. Nowadays, the TabNet model is proving to be more effective than gradient boosting algorithms for classification purposes. Similarly, after Autoencoder feature extraction, the TabNet model proved to be the best when the classification was done. Tables 5.6 and 5.7 compare performance metrics among all the classifier models in UNSW-NB15 and NSL-KDD dataset after performing XGBoost feature extraction.

The ROC curve of classification with TabNet for the combination XGBoost feature selection and Autoencoder feature selection in both datasets is shown in Figures 5.5 and 5.6. The AUC score for TabNet is high, proving it to be the best model. Fig 5.3 and 5.4 shows the comparative analysis among the classification categories: - Before feature selection, XGBoost feature selection, and Autoencoder feature selection. From this graph, we can infer that performance on classification is higher after XGBoost feature selection. Autoencoder feature selection performance results are less than results of classification before feature selection. TCN also proved to provide good accuracy even though not the best, indicating that the classification model also extracts the temporal relations. Causal convolutions, dilations, and skip connections incorporated in the network allow the use of present and just past inputs to predict an output. This allows the model to learn and consider the whole history of inputs.

Threshold	Number of features	Accuracy (%)
0.000	40	79.46
0.000	33	80.14
0.002	31	79.58
0.002	29	80.23
0.003	26	80.31
0.005	24	80.47
0.006	20	80.38
0.006	15	79.76
0.023	11	80.57
0.023	7	81.27
0.065	**4**	**82.10**
0.065	2	81.34

Table 5.4: XGBoost feature extraction with different thresholds in NSL-KDD dataset

Feature extraction technique	Accuracy(%)	Precision	Recall	F1-Score
Autoencoder (UNSW-NB15)	85.32	0.8620	0.8532	0.8477
Autoencoder (NSL-KDD)	75.94	0.8079	0.7594	0.7589
XGBoost (UNSW-NB15)	**90.53**	**0.9116**	**0.9053**	**0.9025**
XGBoost (NSL-KDD)	**82.10**	**0.8568**	**0.8210**	**0.8209**

Table 5.5: Results of Autoencoder and XGBoost feature extraction

5.5 Inference

- Preprocessing with a Bayesian-encoder yields the highest classification accuracy of 90.12% and 79.79% in the UNSW-NB15 and NSL-KDD datasets. This is because the Bayesian encoder could accurately map the relation between a categorical variable and a target variable which was not possible with other encoders.
- Features extracted from XGBoost and Autoencoder model are fed as input to XGBoost classifier, DNN, CNN, TCN, and TabNet. Results indicate that XGBoost outperforms in extracting features efficiently from the data. In structured tabular data, XGBoost gives better results because of its ability to minimize the error and thereby consider relevant features while passing through each subsequent tree sequentially.
- TabNet is proven to give good classification results because of its 2-stage feature transformer and mask that ensures only relevant features are passed to the subsequent blocks. So, it is evident that TabNet performs better than gradient boosting algorithms.
- Promising results from TCN also prove the relevance of temporal features in the data.
- The proposed model, XGBoost feature selection, and TabNet classification algorithm proved to be the best, with a maximum detection rate of 93.02% and 84.22% in UNSW-NB15 and NSL-KDD datasets, respectively.

Classifier	Accuracy(%)	Precision	Recall	F1-Score	ROC AUC score
XGBoost	90.53	0.9116	0.9053	0.9025	0.8975
DNN	78.75	0.8879	0.7875	0.7643	0.7636
CNN	80.94	0.8897	0.8094	0.7917	0.7880
TCN	89.75	0.9108	0.8975	0.8941	0.8886
TabNet	**93.02**	**0.9309**	**0.9302**	**0.9293**	**0.9279**

Table 5.6: Comparative result of classifiers in UNSW-NB15 dataset (Dimensionality-15)

Classifier	Accuracy(%)	Precision	Recall	F1-Score	ROC AUC score
XGBoost	82.10	0.8568	0.8210	0.8209	0.8382
DNN	77.72	0.8064	0.7772	0.7771	0.7920
CNN	83.40	0.8405	0.8340	0.8332	0.8405
TCN	76.77	0.8011	0.7678	0.7675	0.7837
TabNet	**88.35**	**0.8845**	**0.8835**	**0.8807**	**0.8790**

Table 5.7: Comparative result of classifiers in NSL-KDD dataset (Dimensionality-4)

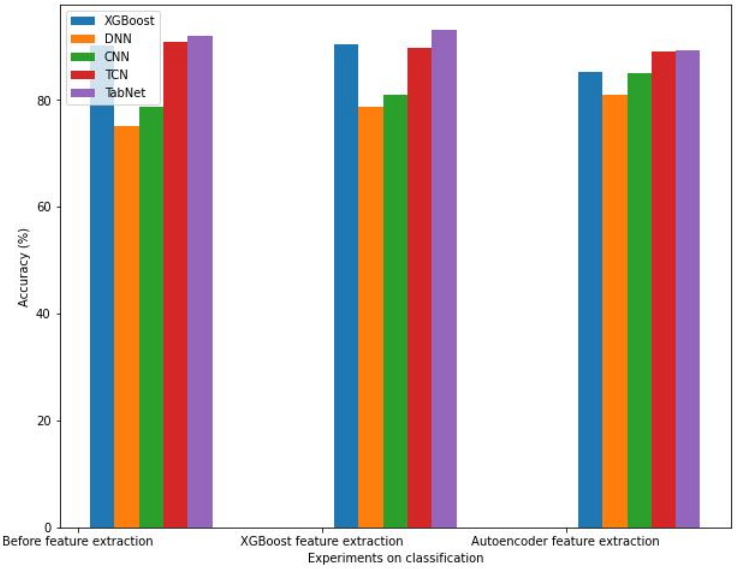

Figure 5.3: Comparison of classification experiments in UNSW-NB15 dataset

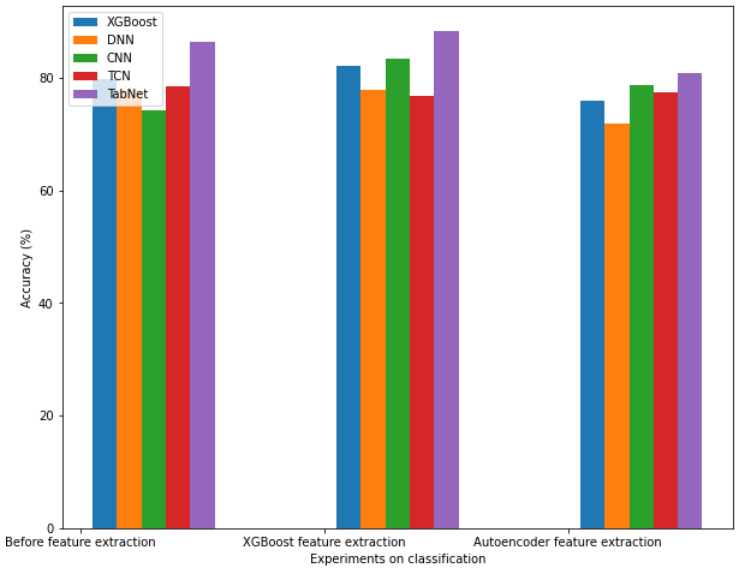

Figure 5.4: Comparison of classification experiments in NSL-KDD dataset

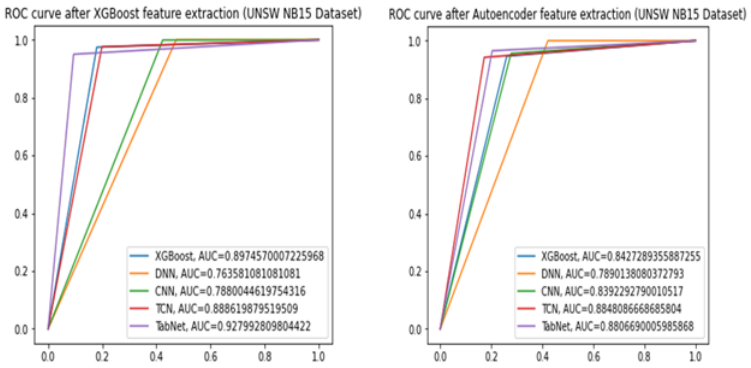

Figure 5.5: ROC curve (UNSW-NB15 dataset)

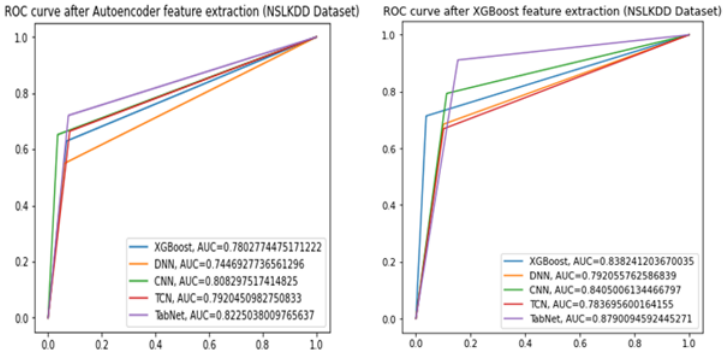

Figure 5.6: ROC curve (NSL-KDD dataset)

Chapter 6

Conclusion

The IDS works on the principle that the features for the conduct of attacks are different from the usual network flow. For proper and efficient classification of attacks, it is necessary to have an effective feature selection technique. In this work, feature selection is carried out with XGBoost and Autoencoder. Classification is done using XGBoost, DNN, CNN, TCN, and TabNet. Preprocessing with a Bayesian-encoder yields the highest classification accuracy of 90.12% and 79.79% in the UNSW-NB15 and NSL-KDD datasets, respectively. The Bayesian encoder could accurately map the relation between a categorical variable and a target variable. Hence further experiments were carried out with a Bayesian-encoded UNSW-NB15 and NSL-KDD dataset. Features extracted from XGBoost and Autoencoder model are fed as input to XGBoost classifier, DNN, CNN, TCN, and TabNet.

Results indicate that Autoencoder-based feature extraction requires more tuning to enhance the detection rate. Hence, XGBoost outperforms in extracting relevant features from the data. In structured tabular data, XGBoost gives better results because of its ability to consider relevant features while passing through each subsequent tree sequentially. TabNet is proven to give good classification results because of its 2-stage feature transformer and mask that ensures only relevant features are passed to the subsequent blocks. So, it is evident that TabNet performs better than gradient boosting algorithms. Promising results from TCN also prove the relevance of temporal features in the data. Finally, a performance comparison is made. The proposed model, XGBoost feature selection, and TabNet classification algorithm proved to be the best, with a maximum detection rate of 93.02% and 84.22% in UNSW-NB15 and NSL-KDD datasets, respectively.

Even though XGBoost outperforms in feature extraction, it takes more time in extracting features corresponding to each threshold. In Autoencoder, training the encoder, decoder, and saving the model with output as 'bottleneck' gives the compressed features. TCN provided good performance results, but it was the only model which provided nearly same accuracy before and after feature extraction. This regards to a smaller number of temporal features after feature extraction. Hence to extract time domain features specifically, some other feature extraction techniques need to be used. This work shall be extended for a more in-depth analysis of temporal features in the future.

References

[1] Devan P, Khare N. An efficient XGBoost–DNN-based classification model for network intrusion detection system. Neural Computing and Applications. 2020 Jan 19:1-6.

[2] Kasongo SM, Sun Y. A deep long short-term memory-based classifier for wireless intrusion detection system. ICT Express. 2020 Jun 1;6(2):98-103.

[3] Moustakidis S, Karlsson P. A novel feature extraction methodology using Siamese convolutional neural networks for intrusion detection. Cybersecurity. 2020 Dec;3(1):1-3.

[4] Alhajjar E, Maxwell P, Bastian N. Adversarial machine learning in network intrusion detection systems. Expert Systems with Applications. 2021 Dec 30;186:115782.

[5] Nguyen MT, Kim K. Genetic convolutional neural network for intrusion detection systems. Future Generation Computer Systems. 2020 Dec 1;113:418-27.

[6] Rajagopal S, Kundapur PP, Hareesha KS. A stacking ensemble for network intrusion detection using heterogeneous datasets. Security and Communication Networks. 2020 Jan 24;2020.

[7] Aslahi-Shahri BM, Rahmani R, Chizari M, Maralani A, Eslami M, Golkar MJ, Ebrahimi A. A hybrid method consisting of GA and SVM for intrusion detection system. Neural computing and applications. 2016 Aug;27(6):1669-76.

[8] Hsu CM, Hsieh HY, Prakosa SW, Azhari MZ, Leu JS. Using long-short-term memory based convolutional neural networks for network intrusion detection. In International wireless internet conference 2018 Oct 15 (pp. 86-94). Springer, Cham.

[9] Gao J, Gan L, Buschendorf F, Zhang L, Liu H, Li P, Dong X, Lu T. Omni SCADA intrusion detection using deep learning algorithms. IEEE Internet of Things Journal. 2020 Jul 14;8(2):951-61.