IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

Comparative Analysis of Intrusion Detection Models on Internet of Vehicles Using TensorFlow Neural Network Classifiers.

[1]Abhishek Sebastian, [1]Pragna R

[1] UG Student(B.Tech)
[1] School of Electronics Engineering
[1] VIT Chennai, Chennai, India.

Abstract: IoV is anticipated to be one of the Future Mobility's ACES (autonomous, connected, electric, and shared) enablers. Security is usually the most crucial concern in the development of IOV since frequent data transmission and intricate connections among many different nodes make hostile attacks more sophisticated and diverse. Many decisions in the IOV architecture are based on artificial intelligence, which is computed utilizing big data and data mining. The attacker's decision could have devastating consequences if they are successful in carrying out a malicious attack and taking complete control of the vehicle. In order to create the best model to identify intrusions from the dataset, various model variations were trained using various optimizers. The best model developed from this study therefore has an accuracy of 99.999897% in detecting typical normal scenarios, 99.27548394% in detecting generic attacks, and 92.0736% in detecting exploits. Researchers are becoming more interested in intrusion detection as a technique that effectively guards the security of IOV by continuously monitoring network data flow, such as network traffic, connections, objects, etc. Systems called intrusion selection (IDS) can spot potentially dangerous substances and threats.

***Index Terms* – Internet of vehicles, Intrusion Detection, Deep-learning, TensorFlow, Machine Learning.**

I. Introduction

A network called the Internet of Vehicles connects vehicles, pedestrians, and various pieces of urban infrastructure. It makes use of a variety of sensors, software, built-in hardware, and connection types to make continuous and dependable communication possible. As a component of a smart city, IoV aims to increase the autonomy, safety, speed, and efficiency of mobility while minimizing resource waste and negative environmental effects. Connectivity is essential to IoV technology. To accomplish this, automakers install the necessary networking and data-gathering hardware, as well as services to process the data and launch specific actions. The essential infrastructure-side digitization takes time to complete. While certain traffic signals, lanes, parking lots, and public transportation stops are connected and offer intelligent assistance to moving vehicles, others continue to operate according to traditional rules [7].

The three primary layers of IoV architecture are as follows:

Perception: It consists of a number of sensors and data collection tools in addition to the hardware required to run the IoV infrastructure. Smartphones, networked street furniture, the global ID terminal, car cameras, and other devices are all part of the perception layer.

Network: This layer is in charge of making connected vehicles visible on the network and sending data to the AI system for processing regarding traffic, road conditions, and driving behaviors. Wi-Fi, 4G/5G, WLAN, Bluetooth, and WAVE are the networks that smart automobiles use the most commonly.

Application: The acquired data is processed, saved, and used by the application layer. It's in charge of recognizing humans and other surrounding linked vehicles in autonomous vehicles and giving instructions to the accelerator, brakes, and engine. Additionally, it is an essential component of all software programmes and services for cars, pedestrians, and commuters [8].

The main objective of the Paper "Intrusion Detection on IOV" are as follows:

- To prepare a data set for machine learning by following specific pre-processing techniques.
- To train an intrusion detection system that can detect intrusions in IOV.

- To analyze the performance of the created model on standard network intrusion datasets UNSW-NB15.

II. Literature Review

T. Janarthanan, S. Zargari, Feature selection in unsw-nb15 and kddcup99 datasets [1], In recent years, network intrusion detection has benefited greatly from the application of machine learning and data mining techniques. Through the use of these techniques, network traffic anomaly detection can be automated. Lack of public data available for research purposes is one of the main issues that researchers are dealing with. Despite certain documented flaws and criticism from a small number of academics, the KDD'99 dataset was utilized by researchers for more than ten years. A new dataset (NSL-KDD), taken from the KDD'99 dataset, was proposed by Tavallaee M. et al. [6] in 2009 in an effort to enhance the dataset so that research on anomaly detection can be conducted using it. The most recently released dataset for intrusion detection research was produced in 2015 and is called UNSW-NB15. This study uses machine learning approaches to analyse the UNSW-NB15 dataset's features and explores important features (the curse of large dimensionality) that can help network systems with intrusion detection. In order to speed up training and testing and conserve resources while maintaining high detection rates, the existing irrelevant and redundant features are removed from the dataset. In this study, a subset of features is suggested, and the results are evaluated in connection to the selection of features in the KDD'99 dataset.

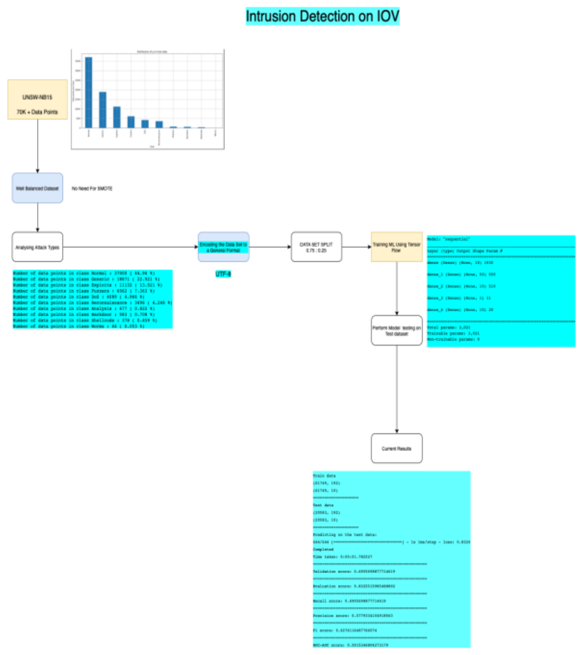
UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set) [2], The lack of a comprehensive network-based data set that can reflect contemporary network traffic scenarios, a wide range of low footprint incursions, and in-depth structured information about the network traffic is one of the main research problems in this area. A decade ago, the benchmark datasets KDD98, KDDCUP99, and NSL KDD were created in order to assess network intrusion detection systems research efforts. Numerous recent studies, however, have demonstrated that these data sets do not accurately reflect network traffic and contemporary low footprint attacks in the context of the current network security environment. This research examines a UNSW-NB15 data set generation to address the problems associated with the lack of network benchmark data sets. This data set combines network traffic assault actions that are now synthesized with real-world modern norms. The UNSWNB15 data set's features are produced using both established and cutting-edge techniques.

UNSW-NB15 Computer Security Dataset: Analysis through Visualization [3],In order to identify any issues with the UNSW-NB25 computer network security or intrusion detection dataset that may need to be addressed by researchers before using this dataset for data-driven model development, such as a machine learning classifier, this paper presents a visual analysis of the dataset. In order to handle typical problems including the removal of redundant features, the conversion of nominal characteristics into numerical representation, and scaling, a number of data preparation algorithms are used for the raw data.

Cybersecurity in Automotive: An Intrusion Detection System in Connected Vehicles [4], This study introduces an embedded intrusion detection system (IDS) for the automotive industry. It operates by using a two-step algorithm to identify potential cyberattacks. A Bayesian network is used to analyse any messages that may be potentially malicious in the first step of the methodology, which filters all messages on the Controller Area Network (CAN- Bus) using spatial and temporal analysis. The analysis provides the likelihood that a given event can be categorized as an attack. The Bayesian networks and the changing status characteristics of the several ECUs over time are both utilized by the detection algorithm to identify a potential assault. Ten state frames are examined in the first step. It determines whether there is or is not a potential assault in the sequence of values using spatial and temporal analysis gleaned from a problem analysis. The presence or absence of an assault is determined in the second step using a Bayesian network that was previously trained using a pre-established data set during the stimulation phase.

Classification Approach for Intrusion Detection in Vehicle Systems [5],The system for detecting intrusions in cars is suggested in this research. To identify DoS and fuzzy assaults, two algorithms—KNN and SVM—are used. Hacking and defensive strategies the dataset for analysis is provided by the research lab. KNN is a non-parametric method designed for classification and regression; it keeps all of the data for the cases that have already been classified and classifies new cases using the same criteria, such as distance functions. SVM is a supervised machine learning technique that can be applied to problems involving classification or regression. K-means and K-medoids are two clustering algorithms that can be used to find the outlier sample and separate it from the rest of the data. Four types of alarms—true positive rate, true negative rate, false positive rate, and false negative rate—can be used to assess the effectiveness of any binary classifier.

III. METHODOLOGY


Intrusion Detection on IOV
UNSW-NB15
70K + Data Points
Well Balanced Dataset
No Need For SMOTE
Analysing Attack Types
Encoding the Data Set to a General Format
UTF-8
DATA SET SPLIT 0.75 : 0.25
Training ML Using Tensor Flow
Perform Model testing on Test dataset
Current Results

```
Number of data points in class Normal : 37000 ( 44.94 %)
Number of data points in class Generic : 18871 ( 22.921 %)
Number of data points in class Exploits : 11132 ( 13.521 %)
Number of data points in class Fuzzers : 6062 ( 7.363 %)
Number of data points in class DoS : 4089 ( 4.966 %)
Number of data points in class Reconnaissance : 3496 ( 4.246 %)
Number of data points in class Analysis : 677 ( 0.822 %)
Number of data points in class Backdoor : 583 ( 0.708 %)
Number of data points in class Shellcode : 378 ( 0.459 %)
Number of data points in class Worms : 44 ( 0.053 %)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type) Output Shape Param #
=================================================================
dense (Dense) (None, 10) 1930

dense_1 (Dense) (None, 50) 550

dense_2 (Dense) (None, 10) 510

dense_3 (Dense) (None, 1) 11

dense_4 (Dense) (None, 10) 20

=================================================================
Total params: 3,021
Trainable params: 3,021
Non-trainable params: 0
```

```
Train data
(61749, 192)
(61749, 10)
====================
Test data
(20583, 192)
(20583, 10)
====================
Predicting on the test data:
644/644 [==============================] - 1s 1ms/step - loss: 0.6326
Completed
Time taken: 0:05:01.782527
==================================================
Validation score: 0.6901698877714619
==================================================
Evaluation score: 0.6325515985488892
==================================================
Recall score: 0.6955498877714619
==================================================
Precision score: 0.5779134184928563
==================================================
F1 score: 0.6274116487744074
==================================================
ROC-AUC score: 0.5915246854213279
```

Fig - 1: Methodology of the system

- Dataset containing 70000+ data points was chosen for training. The Data set used is a variation of the famous UNSW-NB15 Data set.
- In this variation of the data set, it's noted that it's well balanced, hence avoid the use of SMOTE.
- After thorough analysis of the data set, encoded it to a standard UTF-8 format.
- The dataset is now split to 75%/25%, training and validation set.
- TensorFlow is used as the primary ML training platform.
- We perform training in different iterations with changes in the layers, optimization types and input parameters.
- Preparation of a comparison for the best model, will be obtained as the result.

IV. SOFTWARE DEVELOPMENT

4.1 Dataset used:

The raw network packets for the UNSW-NB 15 dataset were synthesized by the IXIA PerfectStorm tool in the Cyber Range Lab of UNSW Canberra in order to present a hybrid of genuine contemporary normal activities and synthetic current attack behaviors. The tcpdump tool was used to capture 100 GB of the raw traffic (e.g., Pcap files). The nine attack categories in this dataset are Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms. The Argus and Bro-IDS tools are used, together with twelve techniques, to generate a total of 49 characteristics with the class label. These features are described in the UNSW-NB15 features.csv file.
A total of two million and 540,044 records can be found in the four CSV files UNSW-NB15 1.csv, UNSW-NB15 2.csv, UNSW-NB15 3.csv, and UNSW-NB15 4.csv.
The ground truth table is UNSW-NB15 GT.csv, while the event list file is UNSW-NB15 LIST EVENTS.csv. This dataset was divided into a training set and a testing set, with the file names UNSW NB15 training-set.csv and UNSW NB15 testing-set.csv, respectively. The testing set consists of 82,332 records from the attack and normal categories, while the training set consists of 175,341 records.

4.2 Software Used:

4.2.1 Google Colab:

Google Research's Colaboratory, also referred to as "Colab," is a product. Data analysis, teaching, and machine learning are three areas where Colab excels. Through the browser, anyone may write and run arbitrary Python code. Colab is a hosted Jupyter notebook service that enables no-cost access to computer resources, including GPUs, and doesn't require any installation [9].

4.2.2 TensorFlow:

TensorFlow is a well-known deep learning and machine learning framework. It was created by the Google Brain Team and made available on November 9, 2015, as a free and open-source library. Python, a programming language used for numerical computations and data processing, is the foundation of machine learning, which makes it faster and more fluid. For a variety of applications, such as video detection, handwritten digit classification, word embedding, natural language processing, and many others, TensorFlow can be used to train and run deep neural networks. TensorFlow may be run on multiple CPUs or GPUs as well as mobile operating systems [10].

ARCHITECTURE OF TENSORFLOW:

Three components make up the TensorFlow architecture:

- Preparation of the data
- Create the model.
- Model training and estimation [11]

Data Preprocessing
01
Model Building
02
Model Training and Evaluation
03

Fig - 2: Architecture of TensorFlow

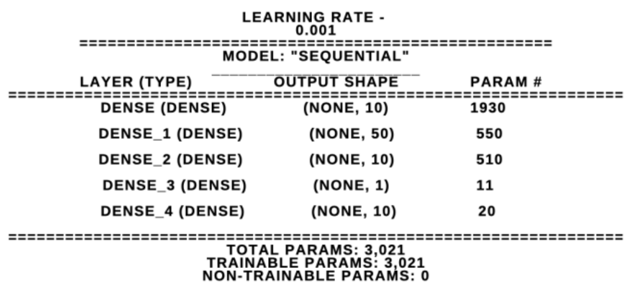
Because it accepts input in the form of multi-dimensional arrays, or tensors, TensorFlow is termed as such. You can draw a "graph" (sometimes referred to as a flowchart) outlining the actions you plan to take in relation to that input. The input enters at one end, moves through this complex system of operations, and then leaves as the output at the other end. TensorFlow got its name because a tensor enters, moves through several computations, and then leaves [12].

V. SOFTWARE DEVELOPMENT

5.1 Analysis on Trained Models:

To prepare the best model to detect network intrusions on the Internet of vehicles, we trained seven different versions of the model to compare its accuracy, performance and time constraints. These models were trained on different numbers of parameters and optimizers to obtain the best model.

Firstly, a **sequential** model was trained based on Adam optimizer with fewer layers, in other words a lesser number of parameters.



```
                    LEARNING RATE -
                         0.001
          =================================================
                  MODEL: "SEQUENTIAL"
          LAYER (TYPE)      OUTPUT SHAPE            PARAM #
=================================================================
          DENSE (DENSE)        (NONE, 10)           1930
          DENSE_1 (DENSE)      (NONE, 50)           550
          DENSE_2 (DENSE)      (NONE, 10)           510
          DENSE_3 (DENSE)      (NONE, 1)            11
          DENSE_4 (DENSE)      (NONE, 10)           20
=================================================================
                  TOTAL PARAMS: 3,021
                TRAINABLE PARAMS: 3,021
               NON-TRAINABLE PARAMS: 0
```

Fig- 3: Design for Model (Adam, "Less #Layers")

The different scores obtained from testing this model, were

1)F1 score: 0.8066759195754676

2)Precision score: 0.7860030995757709

3)Recall Score: 0.8412281980274984

4)Validation score: 0.8412281980274984

From the above scores, it can be inferred that this model has a good recall rate and precisely predicts attacks.

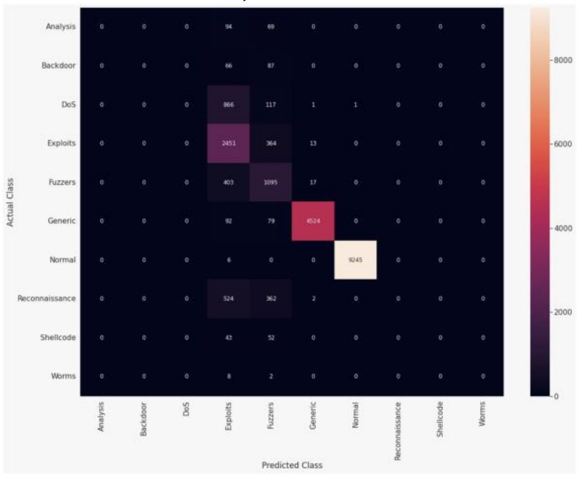
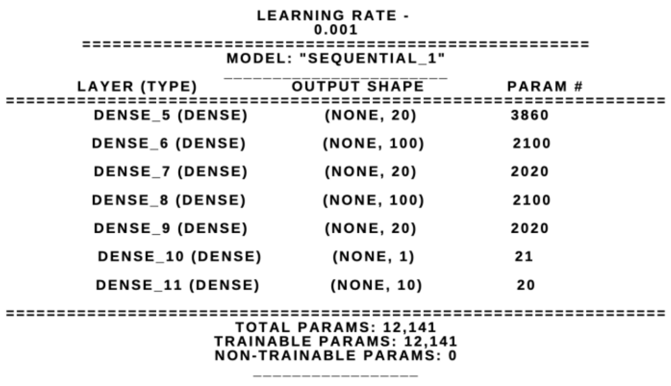


Fig- 4: Confusion Matrix for Model (Adam, "Fewer #Layers")

The Confusion matrix clearly states that the model is able to predict **normal** at a very high accuracy that is 99.999897%, predict can be stated that this model is quite capable of differentiating between a attack and non-attack scenario with very little false-positives.

Moving on to the next model, this model is based on sequential with Adam as it's optimizer and moderately higher number of layers.



```
                    LEARNING RATE -
                         0.001
         =================================================
                MODEL: "SEQUENTIAL_1"
                ______________________
     LAYER (TYPE)          OUTPUT SHAPE          PARAM #
=================================================================
     DENSE_5 (DENSE)        (NONE, 20)            3860
     DENSE_6 (DENSE)        (NONE, 100)           2100
     DENSE_7 (DENSE)        (NONE, 20)            2020
     DENSE_8 (DENSE)        (NONE, 100)           2100
     DENSE_9 (DENSE)        (NONE, 20)            2020
     DENSE_10 (DENSE)       (NONE, 1)             21
     DENSE_11 (DENSE)       (NONE, 10)            20
=================================================================
                  TOTAL PARAMS: 12,141
                TRAINABLE PARAMS: 12,141
                NON-TRAINABLE PARAMS: 0
                  ________________
```

Figure 5: Design for Model (Adam, "Moderate #Layers")

The different scores obtained from testing this model, were

1)F1 score: 0.8109493156041842

2)Precision score: 0.7901173584280121

3)Recall Score: 0.8458922411698975

4)Validation score: 0.8458922411698975

From the above scores, it can be inferred that this model has a good recall rate and precisely predicts attacks.

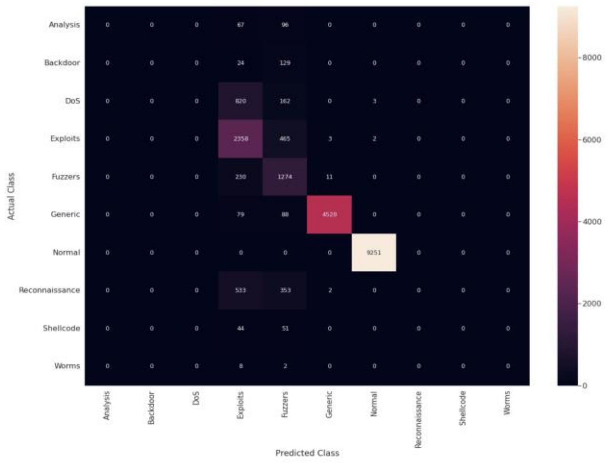
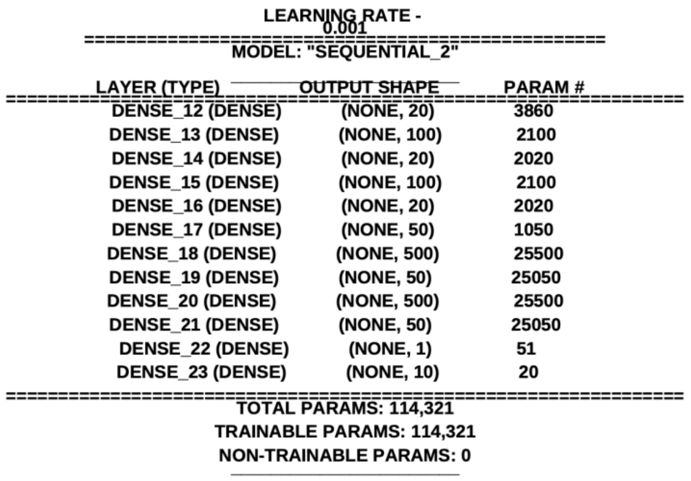


Fig- 6 Confusion Matrix for Model (Adam, "Moderate #Layers")

From the nature of the data set, we can infer that the model trained is capable of detecting an intrusion with great accuracy and with almost 0% false positives.
Moving on to the next model, this model is on **sequential** with **Adam** as its optimizer and very high number of layers.



```
                    LEARNING RATE -
                         0.001
        =================================================
                MODEL: "SEQUENTIAL_2"

     LAYER (TYPE)          OUTPUT SHAPE          PARAM #
=================================================================
     DENSE_12 (DENSE)       (NONE, 20)            3860
     DENSE_13 (DENSE)       (NONE, 100)           2100
     DENSE_14 (DENSE)       (NONE, 20)            2020
     DENSE_15 (DENSE)       (NONE, 100)           2100
     DENSE_16 (DENSE)       (NONE, 20)            2020
     DENSE_17 (DENSE)       (NONE, 50)            1050
     DENSE_18 (DENSE)       (NONE, 500)           25500
     DENSE_19 (DENSE)       (NONE, 50)            25050
     DENSE_20 (DENSE)       (NONE, 500)           25500
     DENSE_21 (DENSE)       (NONE, 50)            25050
     DENSE_22 (DENSE)       (NONE, 1)             51
     DENSE_23 (DENSE)       (NONE, 10)            20
=================================================================
                  TOTAL PARAMS: 114,321
                TRAINABLE PARAMS: 114,321
                NON-TRAINABLE PARAMS: 0
```

Fig-7: Design for Model (Adam, "High #Layers")

The different scores obtained from testing this model, were

1)F1 score: 0.6832239547193361

2)Precision score: 0.6396818480833629
3)Recall Score: 0.7468784919593839
4)Validation score:0.7468784919593839

The sudden decrease in metrics is due to overfitting of the model because of its design. It can be inferred that the best performing model is the one with a moderate number of layers.

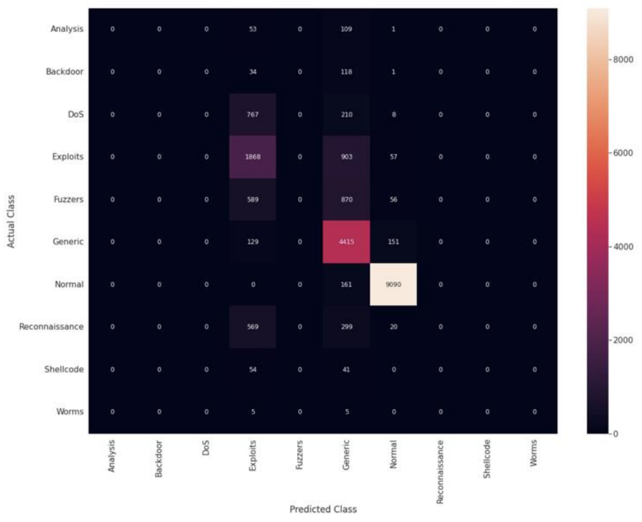


Fig- 8: Confusion Matrix for Model (Adam, "High #Layers")

With that as the base model, different optimizers can be deployed to obtain the most accurate model to detect intrusions. The next model chosen is optimizer retaining the best base mode and changing the optimizer as Adagrad. From the analysis of this model, it can be inferred that this model produces many false positives. Most cases of DoS attacks, Exploits, Fuzzers, Generic and Reconnaissance are detected normal. This can be clearly noticed from the confusion matrix.

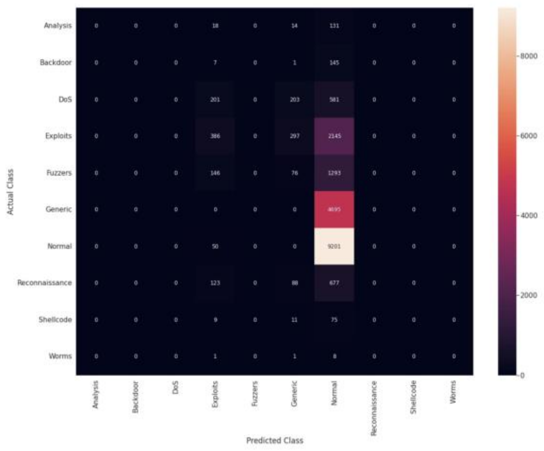


Fig- 9: Confusion Matrix for Model (Adagrad,"Moderate #Layers")

The calculated F1 score is 0.3214107516507874, all the predictions were labelled normal making this model unsuitable for predictions.
It can be concluded that Adagrad is not suitable for analyzing this dataset.
SGD is the optimizer used for the next model. The model structure is not changed, this model had aF1 score: 0.45690997485443 86.
After analyzing this model with the help of a confusion matrix.

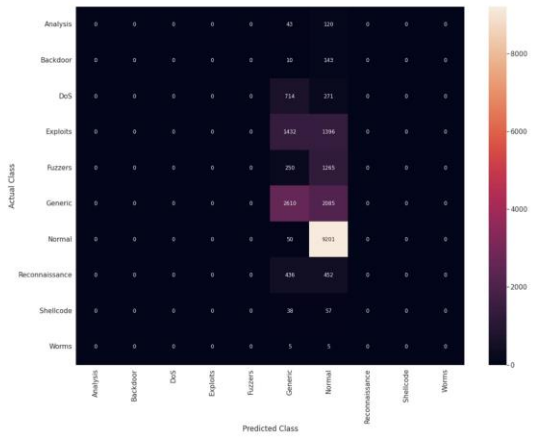


Fig- 10: Confusion Matrix for Model (SGD, "Moderate #Layers") it is concluded that this model also detects more false positives. This prediction of generic attack in this model is more reliable than the previous model. Probability of detecting DoS attacks, Fizzers, Exploits, Reconnaissance is still very low in model.

The best base model with RMSPROP as its optimizer, had significant improvements in differentiating normal cases from other cases resulting in fewer false positives, still the model could not differentiate different types of attack, mostly predicting it as a generic attack.

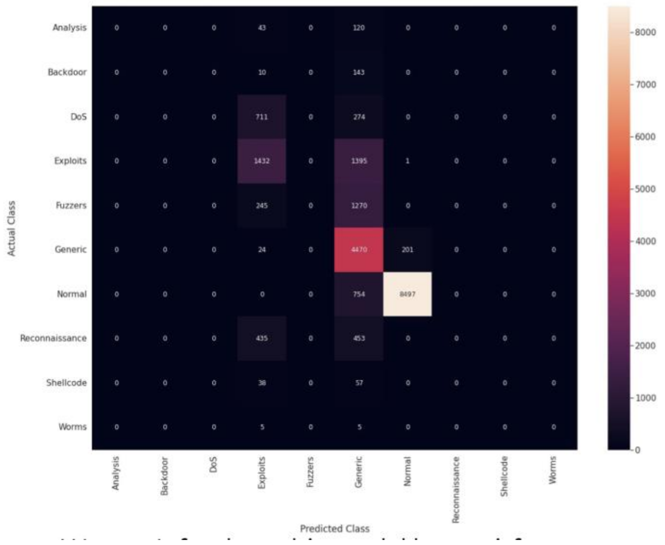


Fig-11: Confusion Matrix for Model (RMSPROP, "Moderate #Layers")
This model had a F1 score: 0.6432438809863271, which is statistically better than other models with SGD and Adagrad.
The next model was trained with Adamax as its optimizer, which is similar to Adam.
There was a significant increase in performance, the F1 score jumped to 0.7772199872847317, which is an increase of 20.31% from the previous trained model. The Confusion matrix clearly depicts that, the model is being able to differentiate other attacks with higher accuracy and precision.

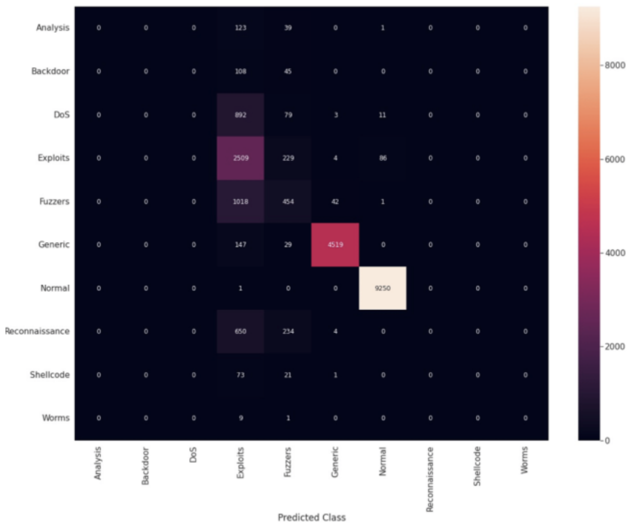


Fig-12: Confusion Matrix for Model (Adamax, "Moderate #Layers")
There are very few predictions of false positives, improved predictions of generic attacks and exploits attacks.
Table -1: Top 5 Ranked Model iterations for detecting network intrusions on internet of vehicles

Rankings	Model	F1-Score
#1	Optimizer: **ADAM** No of layers: Moderate	F1 Score:0.8109
#2	Optimizer: **ADAM** No of layers: Less	F1 Score:0.8066

#3	Optimizer: **ADAMAX** No of layers: Moderate	F1 Score:0.777
#4	Optimizer: **RMSPROP** No of layers: Moderate	F1 Score:0.643
#5	Optimizer: **ADAM** No of layers: High	F1 Score:0.613

From all the above trained models, we can conclude that, a model with moderately high number of layers and Adam as its optimizer is the best suited model for detecting different types of network intrusion attacks.

VI. Conclusion

It can be concluded that from different training methodologies, with dataset that is used in consideration, the trained model with medium number of layers and Adam as it's optimizer has the best performance compared to other model variants. Future works can include to improve higher accuracy for predicting the minor types of attacks like worms, trojan etc.

REFERENCES

[1] Janarthanan, T., & Zargari, S. (2017, June). Feature selection in UNSW-NB15 and KDDCUP'99 datasets. In *2017 IEEE 26th international symposium on industrial electronics (ISIE)* (pp. 1881-1886). IEEE.

[2] Moustafa, N., & Slay, J. (2015, November). UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set). In 2015 military communications and information systems conference (MilCIS) (pp. 1-6). IEEE.

[3] Zoghi, Z., & Serpen, G. (2021). Unsw-nb15 computer security dataset: Analysis through visualization. arXiv preprint arXiv:2101.05067.

[4] Pascale, F., Adinolfi, E. A., Coppola, S., & Santonicola, E. (2021). Cybersecurity in automotive: An intrusion detection system in connected vehicles. Electronics, 10(15), 1765.

[5] Alshammari, A., Zohdy, M. A., Debnath, D., & Corser, G. (2018). Classification approach for intrusion detection in vehicle systems. Wireless Engineering and Technology, 9(4), 79-94.

[6] Tavallaee, M., Bagheri, E., Lu, W., & Ghorbani, A. A. (2009, July). A detailed analysis of the KDD CUP 99 data set. In *2009 IEEE symposium on computational intelligence for security and defense applications* (pp. 1-6). Ieee.

[7] Jin, F., Chen, M., Zhang, W., Yuan, Y., & Wang, S. (2021). Intrusion detection on internet of vehicles via combining log-ratio oversampling, outlier detection and metric learning. Information Sciences, 579, 814-831.

[8] Tuyisenge, Livinus & Ayaida, Marwane & Tohmé, Samir & Afilal, Lissan. (2018). Network Architectures in Internet of Vehicles (IoV): Review, Protocols Analysis, Challenges and Issues: 5th International Conference, IOV 2018, Paris, France, November 20–22, 2018, Proceedings. 10.1007/978-3-030-05081-8_1.

[9] Gunawan, T. S., Ashraf, A., Riza, B. S., Haryanto, E. V., Rosnelly, R., Kartiwi, M., & Janin, Z. (2020). Development of video-based emotion recognition using deep learning with Google Colab. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *18*(5), 2463-2471.

[10]Abadi, M. (2016, September). TensorFlow: learning functions at scale. In *Proceedings of the 21st ACM SIGPLAN International Conference on Functional Programming* (pp. 1-1).

[11]Joseph, F. J. J., Nonsiri, S., & Monsakul, A. (2021). Keras and TensorFlow: A hands-on experience. In *Advanced deep learning for engineers and scientists* (pp. 85-111). Springer, Cham

[12]Goldsborough, P. (2016). A tour of tensorflow. *arXiv preprint arXiv:1610.01178.*