International Journal of Safety and Security Engineering
Vol. 11, No. 3, June, 2021, pp. 231-237
Journal homepage: http://iieta.org/journals/ijsse

Security: Intrusion Prevention System Using Deep Learning on the Internet of Vehicles

Vipparthy Praneeth*, Kontham Raja Kumar, Nagarjuna Karyemsetty

Dept. of Computer Science and Systems Engineering, Andhra University College of Engineering (A), Andhra University, Visakhapatnam, India

Corresponding Author Email: dr.krkumar@andhrauniversity.edu.in

https://doi.org/10.18280/ijsse.110303

Received: 9 April 2021
Accepted: 10 June 2021

Keywords:
a deep neural network, Google CoLab, Internet of Vehicles, network intrusion prevention, security, vehicular network, CICIDS2018 dataset

ABSTRACT

Internet of vehicles supports to transfer of safety-related messages, which help to mitigate road accidents. Internet of vehicles allows vehicle to cooperative communicate, share position and speed data among vehicles and road side units. The vehicular network become prone to large number of attacks including false warnings, mispositioning of vehicles etc. The authentication of messages to identify the normal message packet from attack messages packet and its prevention is a major challenging task. This paper focuses on applying deep learning approach using binary classification to classify the normal packets from malicious packets. The process starts with preparing the training dataset from the open-source KDD99 and CICIDS 2018 datasets, consisting of 1,20,223 network packets with 41 features. The one-dimensional network data is preprocessed using an autoencoder to eliminate the unwanted data in the initial stage. The valuable features are then filtered as 23 out of 41, and the model is trained with structured deep neural networks, then combined with the Softmax classifier and Relu activation functions. The proposed Intrusion prevention model is trained and tested with google Colab, an open platform cloud service, and the open-source tensor flow. The proposed prevention classifier model was validated with the simulation dataset generated in network simulation. The experimental results show 99.57% accuracy, which is the highest among existing RNN and CNN-based models. In the future, the model can be trained on different datasets, which will further improve the model's efficiency and accuracy.

1. INTRODUCTION

The Internet of Vehicles (IoV) plays a vital role in communicating the safety-related message, safeguarding drivers, passengers, people, and the vehicle itself. Other than traditional wired networks have security mechanisms such as gateways, firewalls, and so on, wireless vehicular networks are still vulnerable to safety attacks that threaten nearly the entire infrastructure. VANETs (Vehicular Ad-hoc Network) are sub-division of IoV, operates based on ad-hoc mode, and can be exposed to various sensitive and unsafety actions, including manipulation of communications, send-away, spamming, masquerading, etc. Implementation of security [1] in IoV was found to be one of the significant challenges. To do so correctly and efficaciously, it must comply with the security specifications for protection against attackers and malicious vehicle nodes. To detect intruders is one crucial step in ensuring the security of vehicular networks. Several intrusion detections and prevention methods are available based on statistical analysis, cluster analysis, artificial neural networks, or deep learning. Due to self-learning and the adaptive nature of deep learning, it is a preferable method in intrusion detection and prevention.

Malicious behavior in vehicles network can be detected by connecting and interacting with cameras (CAM), ad hoc vehicle networks, and roadside supporting equipment, and the vehicular node itself. An Intrusion Detection System (IDS) [2] may be considered a safer approach to identifying intruders. The IDS demands that each packet obtained or transferred between vehicle nodes is collected and thoroughly examined. Using test data from the IoV, the protection system can detect normal and even malicious behavior.

Figure 1 illustrates the typical Internet of vehicle architecture [3], which contains vehicles having On-Board Unit (OBU) software integrated with the Intrusion Detection System. In this setup, the vehicular network consists of three normal vehicles with integrated IDS in OBU1 to OBU3 and one vehicle in intruder vehicle OBU4 (represented in red). An open-source Network simulator (NS2 2.34) [4] tool was used to simulate from low to extensive networks, as shown in Figure 2. Simulation of Urban MObility (SUMO) [5] open-source simulator used to generate road structures and network traffic. Various simulation scenario such as low, medium and high density networks were simulated with varying network parameters including size of networks, routing protocols, communication range, packet size etc.

Table 1 shows the output file generated in network simulation which comprise of type of packet, transmission time, packet size, protocol, source and destination address and message etc.

Deep Learning (DL) [6] is a subset of the machine learning (ML) branch that, in effect, shapes the AI branch. DL consists of several hidden neural network layers, one input, and one output layer. This increases efficiency by growing the data collection, i.e., learning a lot more than machine learning with more datasets. Deep learning also automatically considers all miniature features of the Dataset and chooses the most relevant and continuous learning. This also uses different neural

network layers such as convolution, pooling, Softmax, and optimization layers. Each has it is own computing and communication behavior. The order of relationships between different layers is essential. The allocated weights should be optimized after every epoch (iteration) to stabilize the network. The error differences are measured during the analysis phase, and corresponding weights are changed after every round. It helps to improve the learning ability of networks by adaptive nature and tests to be more precise and accurate.

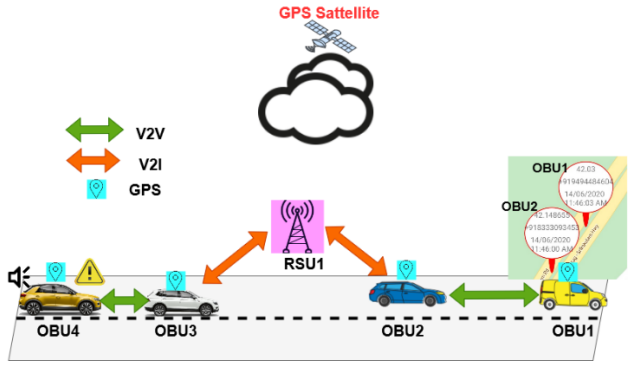

Figure 1. Internet of vehicles architecture

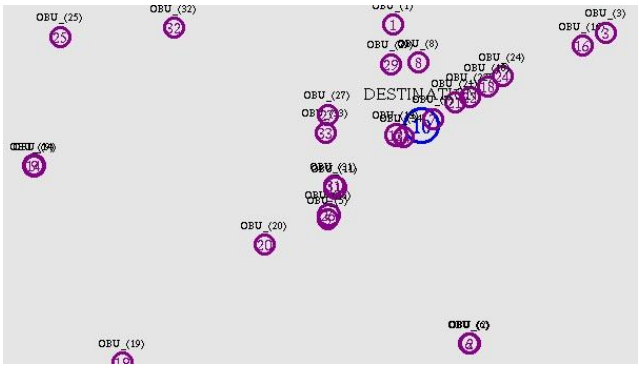

Figure 2. IoV simulation

Deep learning requires further high computation power, where the conventional CPU capacity is not enough. Therefore, dedicated Graphical Processing Unit (GPU). Google has created an electronic platform for writing and executing deep learning and machine learning codes, known as Google Co-laboratory used. Different Python versions and different execution environments are supported. Python can also provide more massive datasets at very high speed from the remote servers to Google cloud storage. After mounting and retrieve the files needed, google co-laboratory provides a high-computing test facility and stores the learning model.

Network intrusion detection is a critical security defense means to protect on-board units and roadside units in the vehicular network. Deep learning for network intrusion detection is an emerging area of recent academic research. Much literature has proposed the successful application of deep learning technology in solving network intrusion detection problems. A deep neural network (DNN) [is a neural network model of deep structure widely used in network intrusion detection. Deep neural networks typically consist of one input layer, multiple hidden layers, and output layers [7]. Refined proposed data from a deep neural network model for the KDD-CUP 99 data collection (DR = 99 percent, FAR = 0.08 percent). A speeded-up deep neural network model with AEs and Softmax layers to complete supervised learning is used to detect network attacks utilized by Dong et al. [8]. The NSL-KDD data collection evaluation accelerated random forest and support vector models by Wang et al. [9], where DR is 97.5% and FAR is 3.5%. LSTM-RNN was mainly part of the recursive neural networks. The intrusion detection system was proposed and used for the NSLKDD data set based on a recursive neural network (DR = 72.95%, FAR = 3.44%, etc.) and traffic flow prediction with Big data analyzed by Lv et al. [10]. Attempted with three classifiers to have an IDS solution. Bat, which was tested on the CICIDS2018 data set and was marginal for most of the results, is based on their correlation. Yin et al. proposed a network intrusion detection system based on a recursive neural network and applied it to the NSLKDD dataset (DR 72.95% percent, FAR 3.44%). In Kim et al.' study, an integrated method based on LSTM-RNN was proposed, and an ADFA dataset was evaluated, resulting in DR being 90% and FAR being 16%. A deep belief network (DBN) is a layered structure of a layer-to-layer restricted Boltzmann machine (RBM). In this paper, to predict the attacks on Network Intrusion Detection System, the use of DNNs is done. The 0.1 rates of learning with DNNs is applied and is the variant of epochs, and for training and benchmarking, the network KDDCup-'99,' Dataset has been used. In this paper, authors developed a new machine learning approach for predicting. Since there were many potential features for network intrusion classification, random forests were used for feature selection based on variable importance scores by Elmasry et al. [11]. The performance of the support vector machine, which used the 14 selected features on the KDD 99 dataset, has been evaluated by comparing it with the total (41) features and popular classifiers.

Table 1. Sample output of network simulation

Event	Time	Node_id	X_Pos	Y_Pos	Pkt_Type	Protocol	Type_msg	Pkt_size
s	0.000168	0	6704.93	6910.51	-99	AGT	DSRCApp	100
r	0.000168	3	6704.93	6910.51	0	RTR	DSRCApp	100
s	0.000548	3	6704.93	6910.51	-99	RTR	DSRCApp	120
s	0.000573	2	6704.93	6910.51	-99	MAC	message	148
r	0.000797	11	6704.93	6910.51	-99	MAC	DSRCApp	120
r	0.000822	8	6704.93	6910.51	0	RTR	DSRCApp	120
r	0.000822	7	6704.93	6910.51	-99	AGT	DSRCApp	120
s	0.002104	6	11478.7	1429.32	0	AGT	message	100
r	0.002104	6	11478.7	1429.32	0	RTR	message	100
s	0.002757	6	11615.1	3812.88	0	AGT	DSRCApp	100
r	0.002757	5	11615.1	3812.88	-99	RTR	DSRCApp	100
s	0.003514	5	15326.7	4352.5	-99	AGT	DSRCApp	100
r	0.003514	14	15326.7	4352.5	-99	RTR	DSRCApp	100
s	0.00373	22	11615.1	3812.88	0	RTR	message	32
s	0.003755	36	11615.1	3812.88	0	MAC	message	60

2. MATERIAL AND TOOLS

2.1 Google Colab [12]

Network intrusion detection can effectively detect the actions and behavior of a vehicular network. The Google Co laboratory is usually referred to as Google Colab, an open-source program that Google offers to anyone with a Google mail account. Google Colab provides Graphical Processing Unit (GPU) and Tensor Processing Unit (TPU). With one runtime, Google Colab provides 32 GB of Random Access Memory (RAM) and 547 GB of disk memory space. This ensures that the GPU software is not used for crypto-currency theft or other illegal purposes. The following configurations provided to co-lab users.

1. None (that indicates the use of personal computer's CPU, no external)
2. Graphical Processing Unit
3. Tensor Processing Unit

After opening a Google Co-lab file, a runtime form is selected for processing.

2.2 Datasets

KDD'99 [13] Dataset was created by DARPA in 1999 using recorded network traffic from the 1999 dataset. It is being pre-processed into 41 features per network connection. Features in KDD'99 Dataset are categorized into four groups, i.e., Basic Features (#1to#9), Content Features (#10to#22), Time based traffic features (#23to#31), and Host-based traffic features (#32to#41). Our project uses the KDD99 open-source Dataset for testing the accuracy of the proposed classifier to classify the normal or attack packet in intrusion detection. There are 4, 94,021 network packets and 41 attributes in the Dataset. There are 23 different output classes with a good network of the "standard" kind, while the other 22 classes reflect different bad connections. Of the total records, the 97.277 (19.69%) were normal, 391.458 (79.24%) DOS, 4.107 (0.83%) R2L and 52 (0.01%) U2R contacts, 1.126 (0.23%) R2L and 52 (0.01%). Table 1 indicates the training class name and its labels and the number of samples included in the training dataset. Description or field of sample data is shown in Table 2.

Canadian Institute for Cybersecurity Intrusion Detection System (CICIDS2018) [14] is a modern anomaly-based NIDS dataset proposed in 2018 and publicly available on the Internet upon request from its ownersCICIDS2018 Dataset contains benign and the most up-to-date common attacks, which resembles the true real-world data (PCAPs). It also includes the results of the network traffic analysis using CICFlowMeter with labeled flows based on the time stamp, source, destination IPs, source and destination ports, protocols, and attack (CSV files). It has 16,000 samples, evenly distributed as follows: Normal (8 000), attack 8,000 of various attacks such as DoS, brute force, port scan, and ping scan. Both KDDCUP 99 AND CICIDS2018 datasets are used to train and test the model. Due to large amount datasets, 1,20,223 data considered for experimentation purpose and Table 3 summarizes the total number of training, testing along with validation dataset.

Table 2. Description of sample data

prtocl_type	ser_type	src_bytes	dst_bytes	su_attempted	num_root	num_shell	error_rate	count	diff_rate	outcome
tcp	http	215	45076	1	1	1	1	0	0	normal.
tcp	http	162	4528	1	2	2	1	1	1	normal.
udp	ftp	236	1228	1	1	1	1	2	0.5	abnormal.
tcp	finger	233	2032	1	2	2	1	3	0.33	normal.
tcp	http	239	486	1	3	3	1	4	0.25	normal.
icmp	http	238	1282	1	4	4	1	5	0.2	normal.
tcp	ftp	235	1337	1	5	5	1	6	0.17	abnormal.
udp	http	234	1364	1	66	6	1	7	0.14	abnormal.
tcp	finger	239	1295	1	7	7	1	8	0.12	abnormal.
tcp	http	181	5450	1	8	8	1	9	0.11	normal.
udp	http	184	124	1	1	1	1	10	0.1	normal.
tcp	http	185	9020	1	2	2	1	11	0.09	normal.
tcp	http	239	1295	1	1	1	1	12	0.08	normal.
udp	login	181	5450	1	2	2	1	13	0.08	normal.
tcp	http	236	1228	1	3	3	1	14	0.07	normal.
icmp	link	233	2032	1	4	4	1	15	0.07	abnormal.
tcp	http	238	1282	1	5	5	1	16	0.06	normal.
icmp	http	235	1337	1	6	6	1	17	0.06	normal.
tcp	name	234	1364	1	7	7	1	18	0.06	normal.
tcp	http	239	486	1	8	8	1	19	0.05	abnormal.
tcp	http	185	9020	1	1	1	1	20	0.05	normal.
tcp	http	184	124	1	2	2	1	21	0.05	normal.
tcp	http	181	5450	1	1	1	1	22	0.05	normal.
tcp	http	239	1295	1	2	2	1	23	0.04	abnormal.
tcp	http	236	1228	1	3	3	1	24	0.04	normal.

Table 3. Dateset

Dataset	**KDD 99**		**CIC-IDS 2018**		**NS 2 Simulation**
Packet type	**Training**	**Testing**	**Training**	**Testing**	**Validation**
Normal	20278	4056	22279	4056	8000
Attack	20458	4092	24912	4092	8000
Total	40736	8148	47191	8148	16000

2.3 Traffic simulator and network simulator

Network simulator (NS2 2.34) simulates transportation using OBUs and RSUs with various dense or sized networks varying from low dense to high dense network with 20 to 300 vehicle nodes and 10 to 30 intruder nodes shown in Figure 2. Intruders are generated various attacks, which are recorded in the log file along with normal packets. An instance of VANET simulation and corresponding trace file generated during simulation process is shown in Figure 2 and Table 1, respectively. Some features of packets generated in the simulation are embedded in packets extracted from the other two sources. An instance of network packets of the simulation is shown in Table 1. Log files are converted into a standard data format to normalize the datasets. These simulation datasets are used to validate the deep learning model in terms of accuracy. Intruder detection in autonomous vehicle or Self driving vehicles [15] simulated in cutting edge tools.

3. PROPOSED INTRUSION PREVENTION SYSTEM

Figure 4 describes the proposed prevention system based on deep neural networks in vehicular networks. Intrusion Detection has to be carried out in three different phases, such as (i) the preprocessing phase, (ii) the feature extraction phase, and (iii) the Classification phase.

In this work, the KDD dataset is chosen, which each data set contains 41 features for each network data packet. Those who have a high potential to engage in intrusions investigation must be picked from these features. There are typical characteristics as a critical contributor to this research work, the detection method, and the decreased number of expected features from the datasets. The number of features used plays a significant role. The key factors behind the decrease in the number of network features are accuracy and processing time.

3.1 Preprocessing

In the preprocessing phase, data filtering and data normalization had done. In this phase, every packet number is fixed between 0.0 and 1.0, using Eq. (1) to normalize the numerical data. Training with structured data ANN is also more productive and used as the best predictor.

$$\text{Z-score} = \frac{\text{Actual value (x)} - \text{mean}(\mu)}{Standard\ deviation(sd)} \quad (1)$$

where, z-score is normalized with a range of 0 to 1, x is the actual value in the Dataset, μ mean, and standard deviation. Such values are used to suit the upper boundaries and lower boundaries of the sigmoid activation function.

3.2 Feature extraction

The main features are chosen for better accuracy, the detection classifier's precision, and the number of false alarms. A mathematical analysis is used to select main characteristics with a high weight and a strong impact based on the POS methodology in the feature selection phase [16]. Conversely, the deletion of a few unnecessary features improves the rate by detection, calculation time, and memory, thus enhancing IDS' overall efficiency. With 13 characteristics added, the time required decreases by 11.4%, and the memory is needed by 27.7%.

3.3 Train the model

The following sample code creates the neural network and adds the 10 nodes to first layer along with shape of input features. Each node functionality or logic is shown in Figure 3.

```
Model = Sequential()
Model = add(Dense(10,input_dim=x.shape[1], activation='relu'))
```

The Classification phase gives the final test results based on the characteristics learned by the self-learning module. Softmax classifier is used as a classification module and its creation was done using statement.

```
Model = add(Dense(y.shape, activation='softmax'))
```

Out of available 41 features of the KDD dataset, only mandatory features are chosen who have high opportunities to engage in intruders. Those feature quantities are symbolic (e.g., protocol). Such characteristics shall be transformed by assigning each function a unique number.

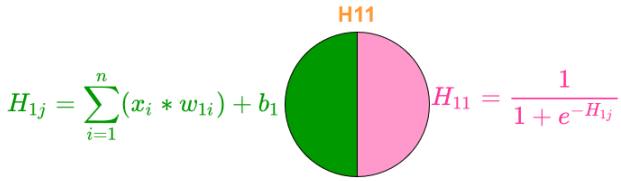

Figure 3. Node logic

MIT Lincoln Labs supplied the Dataset. In this study, ten percent of the training collection, including 494,021 links, used as reported by Lincoln Labs for our method. Our test set consists of the entire labelled sequence of connections, with approximately 4.9 million connections. Hence, able to check program on unexpected connections using the entire Dataset. Rules set are created of six states to correctly classify six separate attack labels up to the current implementation step. From the two attack groups, (the 10 percent training data set) the three top label distributions for attacks: DoS and Probe. Smurf, Neptune, land: DOS attack styles and satan, ipsweep, and portsweep: sample attacks selected.

3.4 Learning mechanism

Using the backpropagation approach to learn the weights,

$$\text{Error}_{\text{tot}} = \sum_{k=1}^{k=2} \frac{1}{2}(\text{target} - \text{output})^2 \quad (2)$$

Error tot is total error computed, target label values and output value.

Update weights, for all weights,

$$\text{W}_{\text{new}} = W_{old} - (\eta * Error_{tot}) \quad (3)$$

where, $\eta\ is\ learning\ rate$.

Learning rate affects the speed of learning, diagnosis behaviors, and stability of the model.

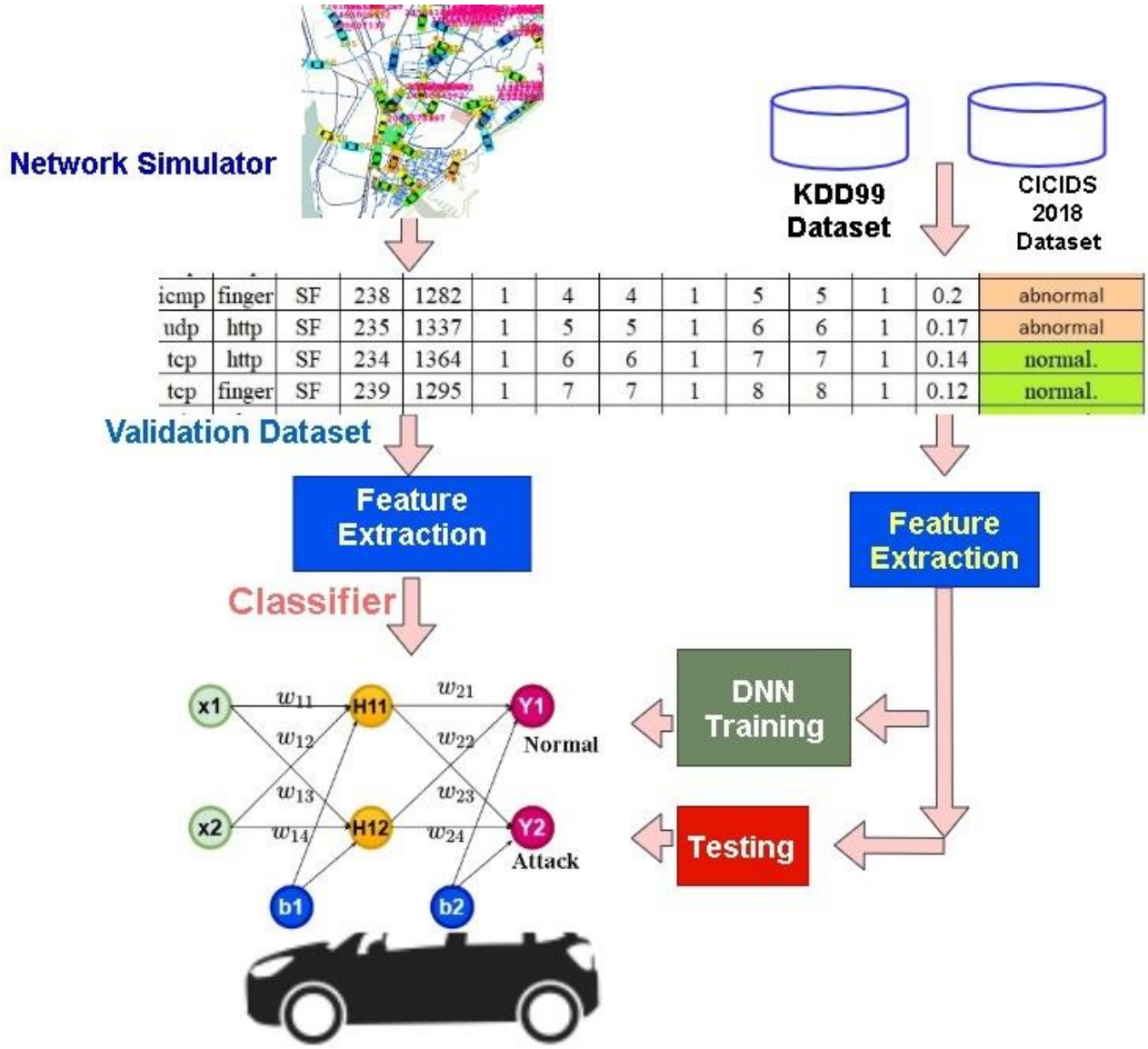

Figure 4. Proposed deep learning-based IPS

3.5 Test and validating the model

The KDD99 generated from a simulation of a military network environment and CICIDS 2018 dataset publicly available Canadian institute of cybersecurity for research and development used to train and test the proposed model.

For training, 80% of both datasets were used, and 20% of the dataset was used to test the model. Network simulator generated Dataset used for validating the proposed model.

The performance of the proposed binary classifier [17] is represented as four outcomes.

True Normal: Ability to predict normal packets as normal.

True Attack: Ability to predict attack packets as an attack.

False Normal: Ability to wrongly predicting the attack packets as normal.

False Attack: Ability to wrongly predicting the normal packets as attack packets.

4. PERFORMANCE EVALUATION

Accuracy indicates the correctness of the (Normal or attack) classifier model to classify the packets over the total number of packets evaluated. The mathematical representation [18] and analysis of accuracy is shown in Eq. (2).

Accuracy = Probability of being correct

= P (1 ∩ Attack) + P (0 ∩ Normal)

Since P (A ∩ B) = P(A|B) P(B), rewrite

P (1 ∩ Attack) = P(1 | Attack) P(Attack)

If the packet is Attack, the probability that the model predicts as Attack is referred to as sensitivity.

P (1 | Attack) p(Attack)

Similarly, P (0 ∩ Normal) = P(0 | Normal) P(Normal)

If the packet is Normal, the Probability that the model predicts as normal is referred to as specificity.

P (0 | Normal) P(Normal)

Accuracy alone not enough to measure the performance of the model. Confusion matrix provides various performance measures such as precision and recall, along with F1-Score, which scores the best model by weighting equal importance to actual and prediction of true and false data. Also developed

and recorded confusion matrices that overcome the accuracy measurement disadvantages, as shown in Table 4.

Table 4. Confusion matrix

Dataset		Model Prediction	
		1	0
Ground Truth	Attack	True Attack (TA)	False Normal (FN)
	Normal	False Attack (FA)	True Normal (TN)

The model predictions [19] can be categorized into four outcomes: a true attack, true normal, false attack, and false normal are various supporting indicators for measuring accuracy. Accuracy tests are used to differentiate the proposition of the normal packets and the attack packets using Eq. (4).

$$\text{Accuracy} = \frac{\text{Total correct predictions}}{\text{Total items participated}} = \frac{(TA+TN)}{(TA+TN+FA+FN)} \quad (4)$$

Accuracy is the classifier's accuracy. The calculation is based on Eq. (4). The intensity of the attack from all samples identified by the test set is correctly marked; out of total attack predictions, many are attacked packets.

$$\text{Precision} = \frac{\text{Actual predicted Attack}}{\text{Total attack predictions}} = \frac{TA}{TA+FA} \quad (5)$$

Eq. (6) used to evaluate the one performance measure of the proposed model is a recall. The recall is the classifier's integrity, the correct labeled attack rate for all attack samples in the test set. It is also sensitivity; that is, out of total actual attacked packets, how many are predicted attacked.

$$\text{Recall} = \frac{\text{Predicted attack}}{\text{Total actual attacks}} = \frac{TA}{TA+FN} \quad (6)$$

F1_score computed using Eq. (7) can be viewed as the harmonic mean of the precision and recall indicators; that is, F1_score helps decide the best model in terms of true and false predictions.

$$\text{F1\_Score} = \frac{2 \text{ X Precision X Recall}}{\text{Precision+Recall}} \quad (7)$$

The above performance metrics are measured to evaluate the correctness of our proposed model in classifying the normal packets as normal and attacked packets as an attack. Detail result analysis and discussions are done in the results and discussion section.

5. RESULTS AND DISCUSSIONS

5.1 Results

As illustrated in Figure 5, the results of several approaches are compared to verify the overall detection performance of the IDS model for intrusion data. The efficiency of the network intrusion detection model depends on the reasonable determination of its evaluation indicators. The higher the exactness, precision, warning, and f-score, the lower the FAR, indicating the classifier's efficiency. The precision and alert of an ideal classification hit 1, and the FAR is 0.

5.2 Discussions

There are 1,20,223 data points and 41 attributes in the Dataset. The data has 23 different output classes, out of which the normal class constitutes a suitable communication link [20], while the other 21 classes represent various types of bad connections. The majority of data points come from the 'natural' group (good connections), around 60.33%. Class "Neptune." (35.594%) and "back" are the highest data points for categories that belong to bad connections (0.665%). The rootkit classes, 'load module.', 'FTP write' and 'multihop' are the most minor data points with fewer than 10 data points per class, and 'spy' has the lowest number. This data set is rather unequalled; therefore, need to build a model that correctly categorizes points belonging to these different groups. Figure 6 shows the comparison evaluation [21] of proposed model and existing model with respect to accuracy, precision, recall and F1-score. On the KDD-CUP 99 data, the precision rate in this paper is 99.57 percent higher than that of 98.62 percent of CNN, and the exactness in literature is 78.24 percent.

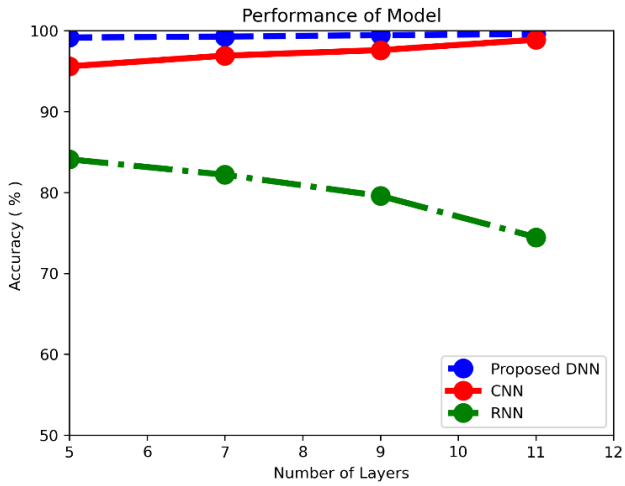

Figure 5. Accuracy of proposed deep learning approach

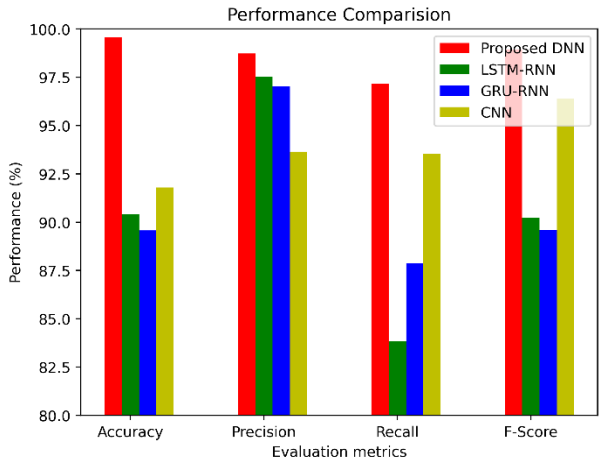

Figure 6. Comparison of evaluation metrics

6. CONCLUSIONS

The Intrusion prevention for the safe and secure transmission of safety packets in Internet of Vehicle, using the KDD99 and CICIDS 2018 datasets, a deep neural network

classifier model was proposed and experimented. DNN parameters using a supervised pre-training approach for deep belief networking with probability-based vectors, followed by the traditional stochastic gradient descent protocol extracted from the vehicle network packets. The proposed model was rigorously trained using CICIDS 2018 and KDD 99 military dataset and validated with simulated data generated using NS2 Network simulator. The DNN gives every class the Probability that normal packets and attack packets are discriminated against to recognize any malicious attacks. Accuracy, precision, recall, and F1-score were evaluated and compared with standard CNN and RNN models. In future, more dataset can be used to train the model to increase accuracy in real time and also some more features can be customized or filtered to reduce the training time.

REFERENCES

[1] Liu, G., Zhang, J. (2020). CNID: Research of network intrusion detection based on convolutional neural network. Journal of Dynamics in Nature and Society (DDNS), 2020: 1-11. https://doi.org/10.1155/2020/4705982

[2] El Boujnouni, M., Jedra, M. (2018). New intrusion detection system based on support vector domain description with information gain metric. International Journal of Network Security, 20(1): 25-34. https://doi.org/10.6633/IJNS.201801.20(1).04

[3] Tuohy, S., Glavin, M., Hughes, C., Jones, E., Trivedi, M., Kilmartin, L. (2015). Intra-vehicle networks: A review. IEEE Transactions on Intelligent Transportation Systems, 16(2): 534-545. https://doi.org/10.1109/TITS.2014.2320605

[4] Network Simulator-2. https://www.isi.edu/nsnam/ns/, accessed on January 12, 2021.

[5] Lopez, P.A., Behrisch, M., Bieker-Walz, L., Erdmann, J., Flötteröd, Y., Hilbrich, R., Lücken, L., Rummel, J., Wagner, P., Wiessner, E. (2018). Microscopic traffic simulation using SUMO. 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 2575-2582. https://doi.org/10.1109/ITSC.2018.8569938

[6] Deep Learning basics. https://www.deeplearning.ai/program/deep-learning-specialization/, accessed on January 12, 2021.

[7] Rawat, W., Wang, Z. (2017). Deep convolutional neural networks for image classification: A comprehensive review. Neural Computation, 29(9): 2352-2449. https://doi.org/10.1162/NECO_a_00990

[8] Dong, H.S., An, K.K., Choi, S.C. (2016). Malicious traffic detection using Kmeans. Journal of Korean Institute of Communications and Information Science, 41(2): 277-284. https://doi.org/10.7840/KICS.2016.41.2.277

[9] Wang, Y., Li, W., Yang, Z. (2017). Network intrusion detection based on random forest and support vector machine. 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), pp. 635-638. https://doi.org/10.1109/CSE-EUC.2017.118

[10] Lv, Y., Duan, Y., Kang, W., Li, Z., Wang, F. (2015). Traffic flow prediction with big data: A deep learning approach. IEEE Transactions on Intelligent Transportation Systems, 16(2): 865-873. https://doi.org/10.1109/TITS.2014.2345663

[11] Elmasry, W., Akbulut, A., Zaim, A.H. (2019). Empirical study on multiclass classification-based network intrusion detection. Computational Intelligence, 35(4): 915-954. https://doi.org/10.1111/coin.12220

[12] Google Colab. http://colab.research.google.com, accessed on May 5 2020.

[13] Hettich, S., Bay, S.D. (1999). The UCI KDD Irvine, CA: The University of California, Department of Information and Computer Science. Available at: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, accessed on May 5 2020.

[14] Canadian Institute for Cybersecurity IDS 2018. https://www.unb.ca/cic/datasets/ids-2018.html, accessed on January 12 2021.

[15] Ali Alheeti, K.M., McDonald-Maier, K. (2016) Hybrid intrusion detection in connected self-driving vehicles. 22nd International Conference on Automation and Computing (ICAC), pp. 456-461. https://doi.org/10.1109/IConAC.2016.7604962

[16] Xiao, Y., Xing, C., Zhang, T., Zhao, Z. (2019). An intrusion detection model based on feature reduction and convolutional neural networks. IEEE Access, 7: 42210-42219. https://doi.org/10.1109/ACCESS.2019.2904620

[17] Yin, C., Zhu, Y., Fei, J., He, X. (2017). A deep learning approach for intrusion detection using recurrent neural networks. IEEE Access, 5(7): 21954-21961. https://doi.org/10.1109/ACCESS.2017.2762418

[18] Yang, H., Wang, F. (2019). Network intrusion detection model based on improved convolutional neural network. Journal of Computer Applications, 39(9): 2604-2610. https://doi.org/10.1109/ACCESS.2019

[19] Zhou, Y., Cheng, G. (2019). An efficient network intrusion detection system based on feature selection and ensemble classifier. http:/arxiv.org/abs/1904.01352.

[20] Chung, J., Gulcehre, C., Cho, K., Bengio, Y. (2015). Gated feedback recurrent neural networks. Proc. 2nd Int. Conf. Int. Conf. Mach. Learn., New York, USA, pp. 2067-2075.

[21] Dong, B., Wang, X. (2016). Comparison deep learning method to traditional methods for network intrusion detection. Proceedings of 8th IEEE International Conference on Communication Software and Networks (ICCSN), China, pp. 581-585. https://doi.org/10.1109/ICCSN.2016.7586590