Anomaly-Based DNN Model for Intrusion Detection in IoT and Model Explanation: Explainable Artificial Intelligence

Bhawana Sharma, Lokesh Sharma, and Chhagan Lal

Abstract IoT has gained immense popularity recently with advancements in technologies and big data. IoT network is dynamically increasing with the addition of devices, and the big data is generated within the network, making the network vulnerable to attacks. Thus, network security is essential, and an intrusion detection system is needed. In this paper, we proposed a deep learning-based model for detecting intrusions or attacks in IoT networks. We constructed a DNN model, applied a filter method for feature reduction, and tuned the model with different parameters. We also compared the performance of DNN with other machine learning techniques in terms of accuracy, and the proposed DNN model with weight decay of 0.0001 and dropout rate of 0.01 achieved an accuracy of 0.993, and the reduced loss on the NSL-KDD dataset having five classes. DL models are a black box and hard to understand, so we explained the model predictions using LIME.

Keywords Intrusion detection system (IDS) · ML · DL · DNN · KNN · SVM · DT · LIME

1 Introduction

In recent years, IoT has been widely used in many fields such as smart cities, healthcare, and automobiles [1]. With the advancement in network technologies, there is growth in connected devices and big data in IoT systems [2], and the network is more prone to attacks. Thus, network security is essential, and there is a need to detect

B. Sharma (✉) · L. Sharma
Department of Information Technology, Manipal University Jaipur, Dehmi Kalan, Jaipur, Rajasthan 303007, India
e-mail: bhawana2104@gmail.com

L. Sharma
e-mail: lokesh.sharma@jaipur.manipal.edu

C. Lal
Department of Intelligent Systems, Cybersecurity Group, TU Delft, Mekelweg 5, Delft 2628, South Holland, The Netherlands

S. Rawat et al. (eds.), *Proceedings of Second International Conference on Computational Electronics for Wireless Communications*, Lecture Notes in Networks and Systems 554,
https://doi.org/10.1007/978-981-19-6661-3_28

the attacks, and measures should be taken to prevent the devices from such attacks [3]. Different types of new attacks are rapidly increasing with the enlargement in network size. Therefore, efficient intrusion detection systems (IDS) are needed to detect intrusions or attacks in the IoT networks.

IDS are classified into two types. One is signature-based IDS in which pre-stored signatures are matched to detect attacks, and another is anomaly-based IDS in which any deviation from normal behavior is identified as attacks [4]. In signature-based IDS, it is difficult to identify new or unknown attacks because it works on matching with predefined attacks, and thus anomaly-based IDS are used for today's network, which detects the attacks based on behavior and can detect new attacks or unknown attacks. In an anomaly-based IDS, there can be false positives as any deviation from normal is classified as an attack, so an efficient technique is needed to reduce the number of false positives. Efficient ML and DL techniques can remove this weakness.

Nowadays, machine learning (ML)/deep learning (DL) techniques are widely used for the computation of large datasets and are providing good results [5]. Thus, researchers are also using ML and DL techniques in the field of cyber security and proposed various models based on ML/DL methods for NIDS in IoT networks such as KNN/SVM and DNN/CNN [6, 7].

ML and DL models are a black box and are hard to understand as they provide only predictions and not the explanation, so the explainable AI concept is introduced, and researchers are working in this field [8, 9]. Models are visualized and explained using LIME. LIME is the most popular method for the explanation of models as it explains the predictions made by the model [10]. In this paper, we proposed DNN-based NIDS, where we reduced the number of features using the filter method and then applied the DNN model for classification and explained the prediction using LIME.

2 Literature Review

In recent years, the field of anomaly-based intrusion detection systems has been drawing the attention of many researchers. In IoT networks, different models based on ML/DL are proposed for IDS, such as SVM/KNN and DNN/CNN. Deep learning techniques have achieved good results in NIDS.

Shone et al. have proposed deep learning and evaluated the model on a publicly available NSL-KDD dataset [11]. Al-Zewairi et al. have proposed the DL model and evaluated it on a publicly available UNSW-NB 15 dataset, and achieved an accuracy of 99% [12]. Alrashdi et al. proposed anomaly-based detection for IoT system: A DIoT using random forest classifier and evaluated the model on the UNSW-NB15 dataset and achieved an accuracy of 99.34% [13]. Xiao et al. proposed a CNN-IDS model and used the KDDCup99 dataset for evaluating the model and achieved an accuracy of 94.0% [14]. Verma et al. proposed a 1D-CNN model and utilized the NSL-KDD dataset for evaluation, and showed an accuracy of 79% and a high detection rate [15]. Ge et al. proposed an FNN model for intrusion detection and

utilized BoT-IoT dataset to train and then evaluate the proposed model for different attack classes, and the multi-class classification model achieved an accuracy above 99% [16].

Fenanir et al. applied different Machine learning-based models on publicly available NSL-KDD and UNSW-NB15 datasets and showed the highest accuracy using a Decision tree (DT) [17].

In [10], Zhou et al. proposed stabilized Lime for model explanation and applied random forest classifier on breast cancer dataset, where the classifier achieved the accuracy of 95% and explained the model using a specific instance of the dataset.

The literature study showed that the different ML and DL techniques are applied to detect the attacks; however, there are certain issues that need to be resolved. The class imbalance issue needs to be solved. The number of features needs to be reduced, which reduces the computation cost. ML and DL models are hard to understand and need explanation methods that explain the predictions of the model.

3 Proposed Framework

We proposed DNN-based NIDS to detect attacks in IoT networks in this paper. We have mainly four phases: data preprocessing (normalization and encoding), feature reduction (selecting the most promising features), feature preprocessing (splitting the dataset), then the last phase is training, and testing model, as shown in Fig. 1.

Dataset description: Researchers are using different publicly available datasets to evaluate the model. NSL-KDD dataset is the standard dataset used for evaluation and is widely used by researchers for NIDS [18]. It contains a total of 41 features, out of which three are symbolic values, the rest are numeric values, and one label shows normal and attacks classes. The label has a total of 23 attack classes, which are then grouped into four main attack classes, namely, Probe, DoS, U2R, and R2L. The total number of records in the dataset is 125972, containing five classes Normal (67,342), DoS (45,927), Probe (11,656), R2L (995), and U2R (52).

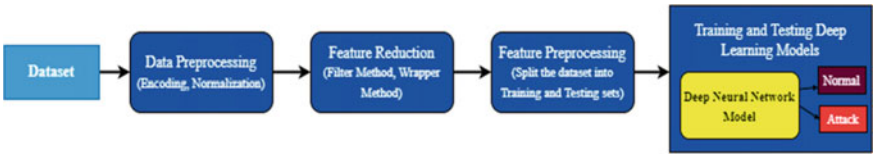

Fig. 1 Workflow of proposed methodology

3.1 Data Pre-Processing

In this phase, features are encoded and normalized. We convert the symbolic features into numeric values for calculation using the label or one-hot encoding. In our experiment, we used label encoding and converted the 3 symbolic features, namely 'protocol type,' 'flag,' and 'service,' into numeric values [19]. Five classes in 'attack label', namely, DoS, Normal, Probe, R2L, and U2R are converted into numeric values 0, 1, 2, 3, and 4 using label encoding. In label encoding, the labels are assigned depending on the alphabetic order.

We normalized the dataset using the Min–Max technique to fit the dataset in the model. The values are normalized within the range of [0, 1] so that the model is not biased to higher values of the dataset. The mathematical Eq. (1) shows the Min–Max normalization, where C is the column values, Cmin is the minimum, Cmax is the maximum values of column, and Cnew is the new value.

$$\text{Cnew} = (\text{C} - \text{Cmin})/(\text{Cmax} - \text{Cmin}) \tag{1}$$

3.2 Feature Reduction

In this phase, we reduce the number of features by selecting the most important features and achieve higher accuracy with minimum numbers of features. By the reduction of features, we can reduce the training time of the model and computational cost. Redundant features are also removed from the dataset to reduce feature size.

In our experiment, we have used the filter method for feature reduction, and we found the correlation between the features and highly correlated features are identified. In our dataset, we applied the Pearson correlation coefficient method, and the correlation value is calculated. The features with a value greater than the threshold value of 0.95 are considered highly correlated, and out of the two features, one is dropped. We dropped six features out of highly correlated features, and the dataset is reduced, containing 36 features.

3.3 Feature Preprocessing

In this phase, after encoding, normalization, and feature reduction, the processed data is transformed into a form that can be fed into the model for training. We divided the dataset into two sets: training and testing. We split the dataset into 75% training set to train the model and 25% testing set for testing the model. The training set is further split into 60% training and 15% validation set.

3.4 Training and Testing

Finally, the processed data is then fed into the model for training and testing. Model is trained using 60% training dataset, and 15% validation dataset is used to validate the model on an unseen dataset. The model detects the normal/attack types during the training phase and calculates the training accuracy. The model is verified using testing data in the testing phase, and then we calculate the testing accuracy. The model consists of dense hidden layers with different numbers of neurons in each layer and the activation function.

Experimental set up

We build our model using the deep learning Keras library and Google Colab, and TensorFlow. We constructed a DNN model using three dense hidden layers of 64 neurons in each layer. Since there are five classes in the dataset, the last layer is fully connected, containing five neurons. We used the ReLU activation function in each dense layer, and in the last layer, we applied a soft-max function, and then the model is compiled, and loss is calculated using sparse categorical cross-entropy and depending upon the loss, we update the weights using Adam optimizer as shown in Fig. 2. We used the NSL-KDD dataset as described above for the experiment and trained the DNN model, and also tuned the model with different hyperparameters. We applied different weight decay values and epochs and compared the accuracy and loss. The model trained on 0.001, 0.0001, 0.00001 weight decay, and 0.01 dropout rate for 50 epochs.

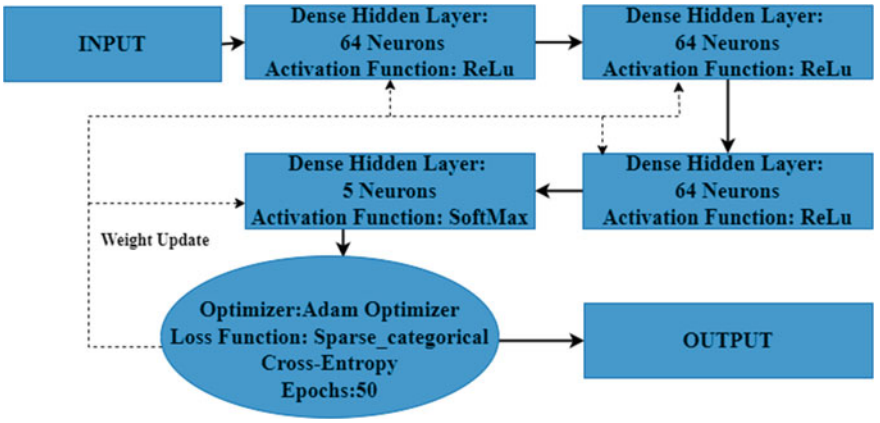

Fig. 2 Architecture of DNN model

4 Evaluation and Analysis

In this paper, we implemented the DNN model for NIDS on Google Colaboratory, evaluated the performance in terms of different evaluation metrics, and compared the model with other ML techniques for classification. We have used the publicly available NSL-KDD dataset for evaluation having numeric and symbolic features. We encoded the dataset using label encoding, and the symbolic features were converted into numeric values using label encoding. The filter-based method is applied for feature reduction, where correlation is calculated between features, and highly correlated features are identified. Then the dataset with reduced features is then applied to our DNN model.

We trained the DNN model with three different weight decay values 0.001, 0.0001, and 0.00001 and the dropout rate of 0.01. The model achieved an accuracy of 99.3% with a weight decay of 0.0001 and a dropout rate of 0.01, and a reduced loss of 0.3.

4.1 Result and Analysis

There are different parameters for model performance. We evaluated the model in terms of the following metrics:

1. **Confusion Matrix**: It is the table where the rows show the true labels and the column shows the predicted labels of the testing dataset. True Positive (TP) is the record count of the attack class correctly classified as the attack class. The record count of the normal class correctly classified as normal is True Negative (TN). False Positives (FP) is the record count of the normal class but is classified as an attack class. The record count of the attack class but classified as normal class is False Negative (FN). The confusion matrix of DNN model is shown in Fig. 3, where the diagonal values show the number of records and the percentage of TP. The model correctly predicted 16,718 records as Normal class, which is 53.08% of total records in the testing dataset.
2. **Accuracy and Loss**: The fraction of records that are correctly predicted/classified as attack and normal class to the total number of predictions is termed accuracy. Figure 4a shows the accuracy of the DNN model trained with three different weight decay values of 0.001, 0.0001, and 0.00001 are 0.985, 0.993, and 0.995, respectively. The error in the predicted and the actual value is termed as loss, and according to the loss, the weights are updated. Loss is less for the DNN model having 0.00001 weight decay, as shown in Fig. 4b. The accuracy of the DNN model is higher for weight decay value 0.00001, but the training time is 382 ms, whereas the model with 0.0001 weight decay has a training time of 315 ms.

Our DNN model with a 0.0001 weight decay and dropout rate of 0.01 achieved the accuracy of 0.993 and reduced loss and 315 ms training time of the model.

We applied different machine learning techniques and compared the accuracy with the proposed DNN model. KNN, decision tree (DT), and support vector machine

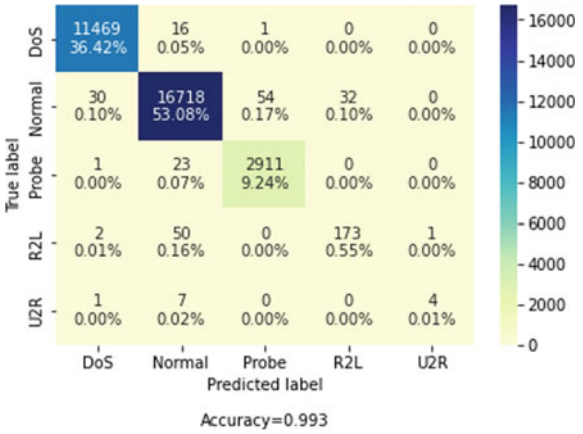

Fig. 3 Confusion matrix of DNN model

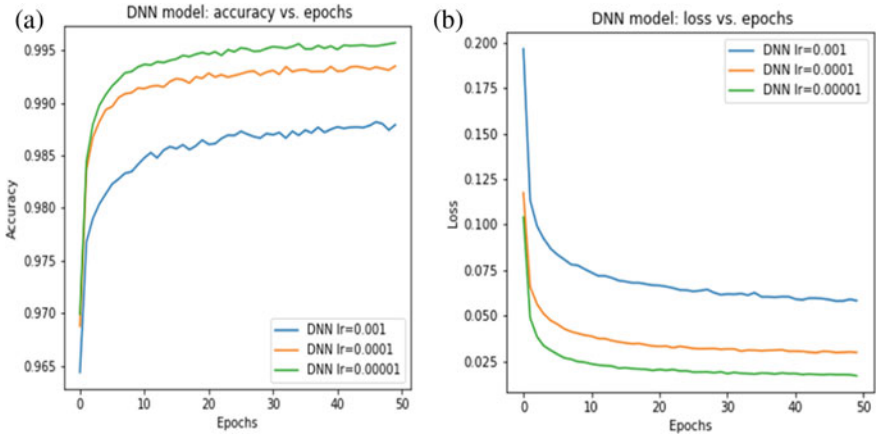

Fig. 4 **a** Accuracy versus epochs **b** Loss versus epochs of DNN having weight decay values $lr = 0.001, 0.0001, 0.00001$

(SVM) techniques are applied to the dataset, and we achieved the accuracy of 0.992, 0.988, and 0.984, respectively. Our proposed DNN model achieved an accuracy of 0.993, having a weight decay of 0.0001, and the training time taken is 315.78 ms, whereas with the weight decay of 0.00001, the accuracy achieved is 0.995, but the training time is 382.72 ms.

5 Model Explanation Using LIME

Machine learning and deep learning models are 'Black boxes' and are hard to understand, so we focus on the explanation of a specific instance. We used LIME for the explanation of the prediction done by the model on the dataset. This verifies that the predictions are the same as actual values.

Instance predicted as normal

We selected the specific instance whose actual value is normal and is predicted as normal, as shown in Fig. 5. On the left of the fig, we see that the model predicted the instance as normal with 99% prediction, and in the center, it shows the features of interest, which helps in the prediction, and in the right, it shows the feature value of the instance. Top ten features are selected for prediction.

Instance predicted as Probe

Similarly, we selected another instance, the actual probe which is correctly predicted as the probe. The model predicted probe with 100% accuracy, as shown in Fig. 6.

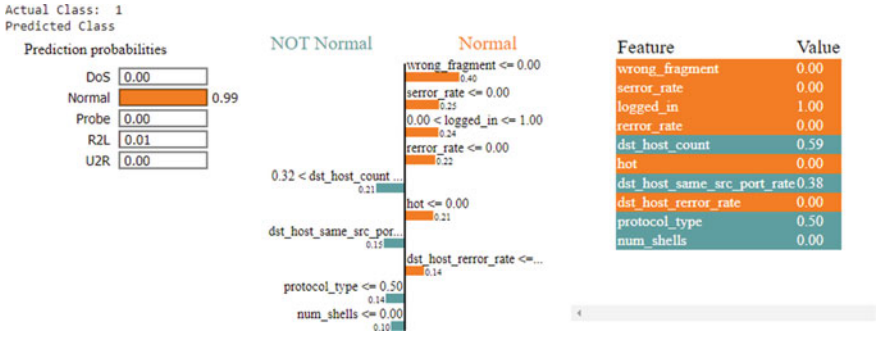

Fig. 5 Normal class predicted as normal

Fig. 6 Probe class predicted as probe

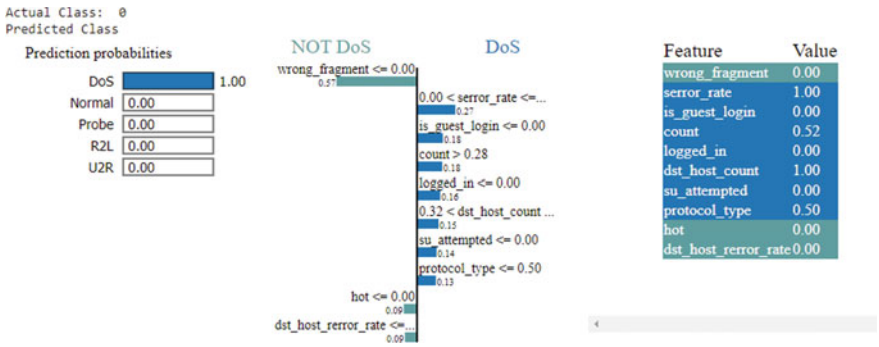

Fig. 7 DoS class predicted as DoS

Instance predicted as DoS

We selected the third instance, which is actual DoS and is predicted as DoS with 100% accuracy as shown in Fig. 7.

6 Conclusion

Nowadays, researchers are seeking interest in intrusion detection systems using machine learning (ML)/deep learning (DL) techniques. In this paper, we proposed a DNN model for intrusion detection, where we reduced the features using the filter method and then tuned the model with the weight decay regularization technique. We compared the model with other machine learning techniques and found that the DNN model achieved the highest accuracy. The model explanation is done using LIME as it is hard to understand the predictions, and for that, we selected three different instances for model verification. Our future work is to remove the class imbalance issue using GANs, reduce the DNN model's training time, and implement the model on real-time IoT systems.

References

1. Da Xu L, He W, Li S (2014) Internet of things in industries: a survey. IEEE Trans Ind Inf 10(4):2233–2243
2. Al-Fuqaha A, Guizani M, Mohammadi M, Aledhari M, Ayyash M (2015) Internet of things: A survey on enabling technologies, protocols, and applications. IEEE Commun Surv Tutorials 17(4):2347–2376
3. Lin J, Yu W, Zhang N, Yang X, Zhang H, Zhao W (2017) A survey on internet of things: architecture, enabling technologies, security and privacy, and applications. IEEE Internet Things J 4(5):1125–1142

4. Sharma B, Sharma L, Lal C (2019) Anomaly detection techniques using deep learning in iot: a survey. In: 2019 international conference on computational intelligence and knowledge economy (ICCIKE), pp 146–149. https://doi.org/10.1109/ICCIKE47802.2019.9004362
5. Chaabouni N, Mosbah M, Zemmari A, Sauvignac C, Faruki P (2019) Network intrusion detection for iot security based on learning techniques. IEEE Commun Surv Tutorials 21(3):2671–2701
6. Ahmad Z, Shahid Khan A, Wai Shiang C, Abdullah J, Ahmad F (2021) Network intrusion detection system: a systematic study of machine learning and deep learning approaches. Trans Emerg Telecommun Technol 32(1):4150
7. Al-Garadi MA, Mohamed A, Al-Ali AK, Du X, Ali I, Guizani M (2020) A survey of machine and deep learning methods for internet of things (iot) security. IEEE Commun Surv Tutorials 22(3):1646–1685
8. Ribeiro MT, Singh S, Guestrin C (2016) "Why should i trust you?" Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining, pp 1135–1144
9. Samek W, Wiegand T, Muuller K-R (2017) Explainable artificial intelligence: understanding, visualizing and interpreting deep learning models. arXiv: preprint arXiv:1708.08296
10. Zhou Z, Hooker G, Wang F (2021) S-lime: stabilized-lime for model explanation. In: Proceedings of the 27th ACM SIGKDD conference on knowledge discovery & data mining, pp 2429–2438
11. Shone N, Ngoc TN, Phai VD, Shi Q (2018) A deep learning approach to network intrusion detection. IEEE Trans Emerg Top Comput Intell 2(1):41–50. https://doi.org/10.1109/TETCI.2017.2772792
12. Al-Zewairi M, Almajali S, Awajan A (2017) Experimental evaluation of a multi-layer feed-forward artificial neural network classifier for network intrusion detection system. In: 2017 international conference on new trends in computing sciences (ICTCS). IEEE, pp 167–172
13. Alrashdi I, Alqazzaz A, Aloufi E, Alharthi R, Zohdy M, Ming H (2019) Ad-iot: Anomaly detection of iot cyberattacks in smart city using machine learning. In: 2019 IEEE 9th annual computing and communication workshop and conference (CCWC). IEEE, pp 0305–0310
14. Xiao Y, Xing C, Zhang T, Zhao Z (2019) An intrusion detection model based on feature reduction and convolutional neural networks. IEEE Access 7, pp 42210–42219
15. Verma AK, Kaushik P, Shrivastava G (2019) A network intrusion detection approach using variant of convolution neural network. In: 2019 international conference on communication and electronics systems (ICCES). IEEE, pp 409–416
16. Ge M, Syed NF, Fu X, Baig Z, Robles-Kelly A (2021) Towards a deep learning-driven intrusion detection approach for internet of things. Comput Netw 186:107784
17. Fenanir S, Semchedine F, Baadache A (2019) A machine learning-based lightweight intrusion detection system for the internet of things. Rev d'Intelligence Artif 33(3):203–211
18. NSL-KDD|dataset (2009) https://www.unb.ca/cic/datasets/nsl.html. [Online; Accessed 19 Oct 2021]
19. Sharma B, Sharma L, Lal C (2022) Feature selection and deep learning technique for intrusion detection system in iot. In: Proceedings of international conference on computational intelligence. Springer, pp 253–261