Anomaly Based Network Intrusion Detection for IoT Attacks using Convolution Neural Network

Sharma, Bhawana ; Sharma, Lokesh ; Lal, Chhagan

DOI
10.1109/I2CT54291.2022.9824229

Publication date
2022

Document Version
Final published version

Published in
Proceedings of the 2022 IEEE 7th International conference for Convergence in Technology (I2CT)

Anomaly Based Network Intrusion Detection for IoT Attacks using Convolution Neural Network

Bhawana Sharma
Dept. of IT
Manipal University Jaipur
Jaipur, India
bhawana2104@gmail.com

Lokesh Sharma
Dept. of IT
Manipal University Jaipur
Jaipur, India
lokesh.sharma@jaipur.manipal.edu

Chhagan Lal
Department of Intelligent Systems
Cybersecurity Group, TU Delft
Netherlands
chhagan.iiita@gmail.com

2022 IEEE 7th International conference for Convergence in Technology (I2CT) | 978-1-6654-2168-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/I2CT54291.2022.9824229

Abstract—**IoT is widely used in many fields, and with the expansion of the network and increment of devices, there is the dynamic growth of data in IoT systems, making the system more vulnerable to various attacks. Nowadays, network security is the primary issue in IoT, and there is a need for the system to detect intruders. In this paper, we constructed a deep learning CNN model for NIDS and utilized the NSL-KDD benchmark dataset, consisting of four attack classes, for evaluating the model's performance. We applied the filter method for feature reduction where highly correlated features are dropped. Our 2D-CNN model achieved an accuracy of 99.4% with reduced loss. We also compared the performance of DNN and CNN models in terms of accuracy and other evaluation metrics.**

Index Terms—**Intrusion Detection System, ML, DL, DNN, CNN, NIDS, HIDS, SVM**

I. Introduction

IoT is gaining popularity nowadays and is extensively used in many fields such as health systems, transportation, etc. IoT is the interconnection of physical objects termed as things, such as actuators/sensors. The large number of devices connected to the network, having limited computational power and storage, makes the network more prone to attacks. Different types of attacks are increasing with the increase in the network, and thus the challenge is to identify the attacks. Since IoT networks are more vulnerable to attacks, there is a need to detect and prevent the system from attacks, making network security and privacy the primary concern in IoT systems [1] [2]. Deep Learning (DL)/ Machine Learning (ML) techniques are considered suitable for computing a large amount of data; thus, ML and DL models are highly used for computation in IoT networks [3] [4]. Nowadays, researchers have proposed various frameworks for NIDS (network intrusion detection systems) using different ML/DL techniques. The key difference between ML and DL techniques is that before applying the ML technique, we select the features from the dataset using different feature selection methods, whereas deep learning techniques have the inbuilt advantage of feature extraction. Different ML techniques are used to detect intrusions such as KNN, SVM, and Deep learning techniques such as DNN and CNN.

An Intrusion detection system identifies the intruder or attack in the network, and then the system administrator takes the preventing measures for the network being attacked by the intruder. IDS is similar to a classification problem where the data is classified as an attack or normal class. IDS are classified as NIDS and HIDS; when the attack or intruder is within the network, it is termed as NIDS, and when it is within the host, it is termed as HIDS [5] [6]. In this paper, we focused on NIDS.

IDS techniques are classified as:

1) Signature Based IDS consists of pre-stored patterns and signatures. The attack or intruder is detected on matching with patterns and signatures. This technique is used to effectively identify known attacks but is inefficient to identify unknown attacks.
2) Anomaly Based IDS analyse the system's behavior, and the activity which is different from the normal activity is considered as an anomaly. The advantage of the technique is that it can detect new or unknown attacks but there can be false positives also. [15].

This paper proposed an anomaly-based intrusion detection system based on a deep learning technique and compared the results with other techniques. We reduced the features using the filter-based correlation method, and then DNN and CNN models classified the data as normal or attack class.

II. LITERATURE REVIEW

In the past few years, researchers working in the area of intrusion detection systems (NIDS) for IoT networks have implemented models using ML and DL techniques. DNN and CNN have achieved remarkable improvement in the field of deep learning techniques. There are various papers on ML and DL-based anomaly detection for the intrusion detection system.

In [7], Shone et al. have proposed a deep learning model to detect intruders, and the model was evaluated on a publicly available NSL KDD dataset. The author used autoencoders for feature learning and then applied the random forest technique for classification.

In [8], Al-Zewairi et al. build a model based on the DL technique for network intrusion detection. The author constructed the model using 5 hidden layers, where each layer had 10 neurons. The model was evaluated on a publicly available UNSW-NB 15 dataset and trained for 10 epochs and also applied 10-fold cross-validation and achieved an accuracy of about 99%.

In [9], Alrashdi et al. proposed anomaly-based detection for IoT system: ADIoT; the experiment was conducted on the UNSW-NB15 dataset. The author trained the model after selecting 12 features from the dataset, then classified normal and anomaly class the Random Forest (RF) algorithm, and achieved 99.34% accuracy.

In [10], Xiao, Y et al. build a CNN-IDS model consisting of pooling layer and convolution layers; the model was evaluated on the publicly available KDDCup99 dataset. The model was trained for 50 epochs and tuned with different parameters; with a 0.3 dropout rate, the model achieved an accuracy of 94.0%.

In [11] proposed 1D-CNN model consists of a convolution layer of 64 filters of 5 size, a pooling layer of size 2, and at last classification dense layer of 128 neurons with a dropout rate of 0.5. The author utilized the NSL-KDD dataset, trained the model for 500 epochs, and showed an accuracy of 79% with a high detection rate.

In [12], Ge, M., Syed et al. proposed an FNN model for intrusion detection and trained on BoT–IoT dataset. The author constructed the model for four-class classification with 2 dense hidden layers where each layer is composed of 512 neurons and 4 neurons in the final output dense layer. The author tuned the model using different parameters such as number of neurons and hyperparameters such as learning rates for tuning the model. FNN model achieved an accuracy above 99% for both binary and multi-class classification.

III. PROPOSED FRAMEWORK

In this paper, we proposed a NIDS framework to detect intruder or attack classes in IoT networks using deep learning techniques. We included four main steps: Collection of data and data pre-processing, feature selection using different methods, feature pre-processing for the transformation of the data usable by training model, at last step, train and test the proposed DL model as shown in Fig. 1.

- **Data Pre-processing:** In this step, we collect the dataset, and then the features are encoded using the label or one-hot encoding techniques. We convert the features of the categorical value into integer values and remove redundant data from the dataset. Data is normalized using the min-max technique to fit in our model.
- **Feature Selection:** In this step, we select the features from the dataset, and thus with the reduced number of features, we train the model. We decrease the number of features, thereby reducing the model's storage and computational cost. We can apply different feature selection techniques such as filter, wrapper, and embedded methods for selecting the features.
- **Feature pre-processing:** In this step, the processed data after encoding, scaling, and selecting features are then transformed into the form usable by the training model. Then the processed dataset with selected features is divided into two sets of 75%-25%, named as training set and testing set, respectively. The training set is used to train the model and then verified using the testing dataset.
- **Training and Testing:** In this step, we train and test the model. Training set data is fed into the DL model in the training phase, which detects the normal or attack categories and calculates the trained accuracy. Testing set data is fed into the model in the testing phase to verify the model and calculate the testing accuracy.

IV. EVALUATION AND ANALYSIS

In this paper, we evaluated the model performance using the above steps of intrusion detection. Our proposed CNN-2D model for NIDS is evaluated and compared with other models. We have used the publicly available NSL-KDD dataset for evaluation.

- **Data Description: NSL-KDD dataset** Out of the different publicly available intrusion detection system (IDS) datasets, the NSL-KDD dataset is widely used dataset by researchers to evaluate the models [16]. KDD Cup contains redundant data which are removed in the NSL-KDD cup dataset. Out of different benchmark datasets, researchers extensively use the NSL-KDD cup dataset for the NIDS.

TABLE I
ATTACK CLASSES OF DATASET

Attack_Class	**Attack_SubType**	**No. of Records**
Denial of service (DoS).	”apache2”,”land”,”worm”, ”pod”,”smurf”, ”udpstorm”, ”processtable”,”neptune”,”back”, ”mailbomb”,”teardrop”	45927
Probe.	”ipsweep”,”mscan”, ”portsweep”,,”satan”, “nmap”,”saint”	11656
Root to local (R2L).	”ftp_write”,”named”,”Snmpgetattack”, ”imap”,”phf”,”sendmail”, ”snmpguess”,”httptunnel”, ”warezclient”,”warezmaster”, ”guess_passwd”,”xlock”, ”multihop”,”spy”,”xsnoop”	995
User to root (U2R).	”buffer_overflow”,”perl”, ”ps”,”sqlattack”,”xterm”, ”loadmodule”,”rootkit”	52
Normal		67342
Total Records		125972

  NSL-KDD contains 41 features with 37 numeric values and 3 nominal values, and one label showing the normal/attack category. The dataset contains a total of 23 classes, including 22 attack types and one normal class, which are grouped into 4 main attack classes, namely DoS, Probe, U2R, and R2L, as shown in table I.
- **Data pre-processing:** We uploaded the NSL-KDD dataset stored in the CSV file to Google's Colaboratory. Before training the model, we process the data. Our dataset contains categorical values, so we have to convert the values into numerical values. We divided data pre-processing into two steps:
  1) Encoding: categorical features in the dataset are converted to numeric values using label encoding to make them usable by the training model. NSL-KDD

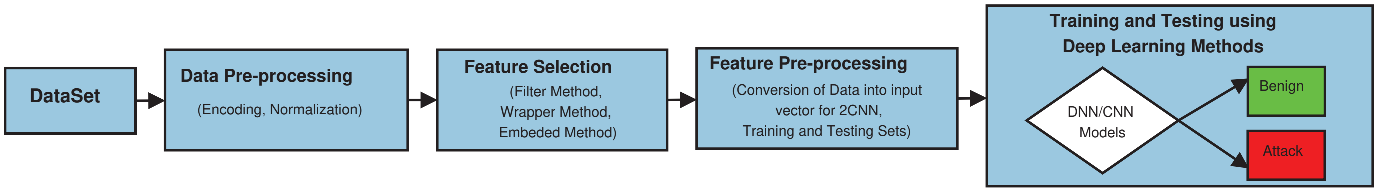

Fig. 1. Workflow of Proposed Methodology

contains 4 categorical data 'flag','service' ,'protocol type', 'attack label' having 11,70,3,5 values respectively. 'protocol_type' contains 3 types ICMP ,TCP, UDP which are converted to 0,1, 2 values using label encoding.

2) Data normalization: we normalized the dataset to bring the values within the range of [0, 1] so that our model should not bias towards the higher values. We normalized Min-Max normalization according to Eq. 1.

$$F_{new} = \frac{F - Min(F))}{Max(F) - Min(F))} \tag{1}$$

Where minimum value and maximum value of the F feature is Min(F) and Max(F) respectively.

- **Feature Selection**:After encoding and normalization of the dataset, we reduced the feature using the filter method. We applied the Pearson correlation filter method for feature selection [14]. The features correlation is found, and the correlation map of the features are derived where the highly correlated features are selected, then the selected features are dropped. Features with a correlation value of 0.95 or greater are considered highly correlated, and one feature is dropped.

  After applying the Pearson correlation filter method on NSL-KDD Cup dataset, we found that 6 features 'dst_host_serror_rate', 'num_root', 'srv_rerror_rate', 'dst_host_srv_rerror_rate','srv_serror_rate', 'dst_host_srv_serror_rate' are dropped from the dataset. After dropping the new dataset contains 36 features.
- **Feature pre-processing**: After encoding and normalization of the NSL-KDD dataset, we split/divide the dataset into three sets: 60%-15%-25% training set, validation set, and testing set containing 36 features. Each record of the dataset having 36 features is transformed into a 6x6 matrix and is then fed into the 2D-CNN model. For the 1D CNN model, each record is transformed into a 36x1 matrix and then the record is fed into the 1D-CNN model. CNN model architecture with convolution, pooling layers is shown in Fig. 2.
- **Experimental setup**: We implemented our model using the deep learning Keras library on Google's Colaboratory and used TensorFlow. CNN model contains an input layer, Convolution layers having filters, pooling layers, and the last layer contains a fully connected layer for classification. For the 1D-CNN model, data is given in the form of an input vector of size Sx1, where S is the number of features. For the 2D-CNN model, data having S features are transformed into an input vector of size MxM, where S is the perfect square of M. If S is not a perfect square, then data is padded with 0.

  Our CNN model consists of three convolution layers with 64, 32, 32 neurons, and each layer has a kernel size of (3x3) with ReLu activation function and two max-pooling layers of (2x2) pool size. The 1D-CNN model kernel size is 3, and Max pooling is size 2. The last layer consists of a fully connected layer with a softmax activation function and 5 neurons depending on the number of classes, as shown in Fig. 3. Model is compiled and Adam optimizer is used to update the weight, sparse categorical cross-entropy loss function is applied. We tuned the model with different hyperparameters such as kernel size and epochs.

  We applied the DNN model [13] consisting of dense hidden layers. The model was constructed using 3 dense hidden layers, and every dense layer has 64 neurons, whereas the last dense layer contains 5 neurons. ReLU activation function is applied on dense hidden layers and the softmax function on the last layer. Adam optimizer is used for weight updating, and the loss function applied is a sparse categorical cross-entropy.

  Various hyperparameters are used for tuning the DNN model, such as drop out, weight decay, learning rate, and epochs. We trained the model with a weight decay of 0.0001, a learning rate having value 0.001, number of epochs is 20, and a dropout rate of 0.01. We tested the model with a test dataset. The accuracy achieved by the DNN, 1D CNN, 2D CNN models are 0.993, 0.992, and 0.994, respectively.

- **Results and Analysis**: There are different criteria for a model's performance; in this paper, we evaluated our deep learning models using the following evaluation metrics, where X is anomaly and Y is normal:
  - True positives (TP): It is the number of X samples that are correctly classified as X.
  - True negatives (TN): It is the number of Y samples that are correctly classified as Y.
  - False positives (FP): It is the number of Y samples that are classified as X.
  - False negatives (FN): It is the number of X samples that are classified as Y.

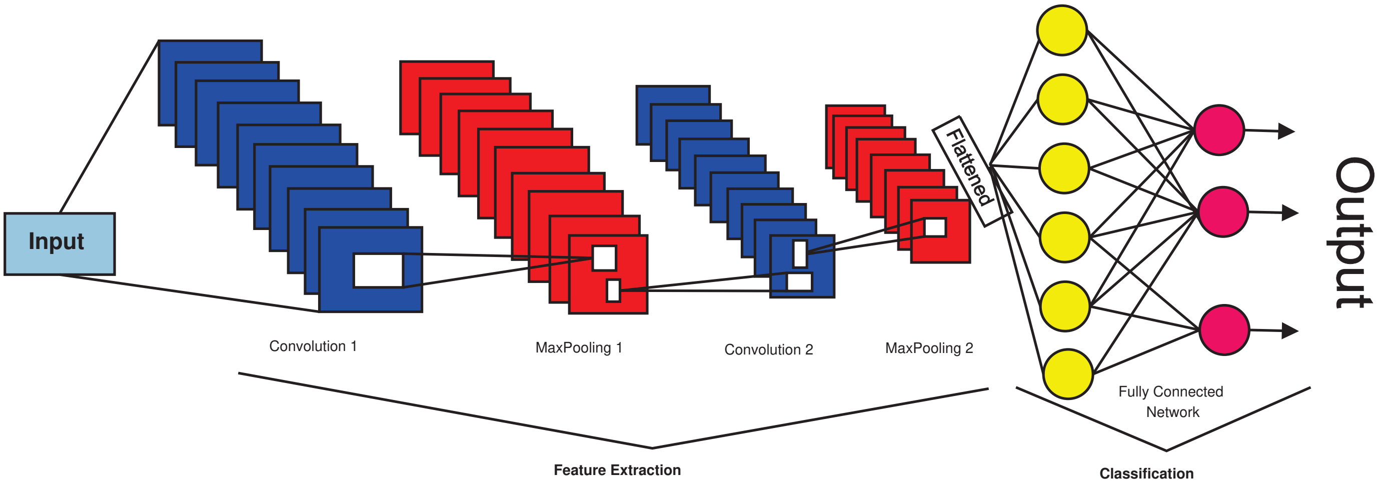

Fig. 2. Architecture of CNN model

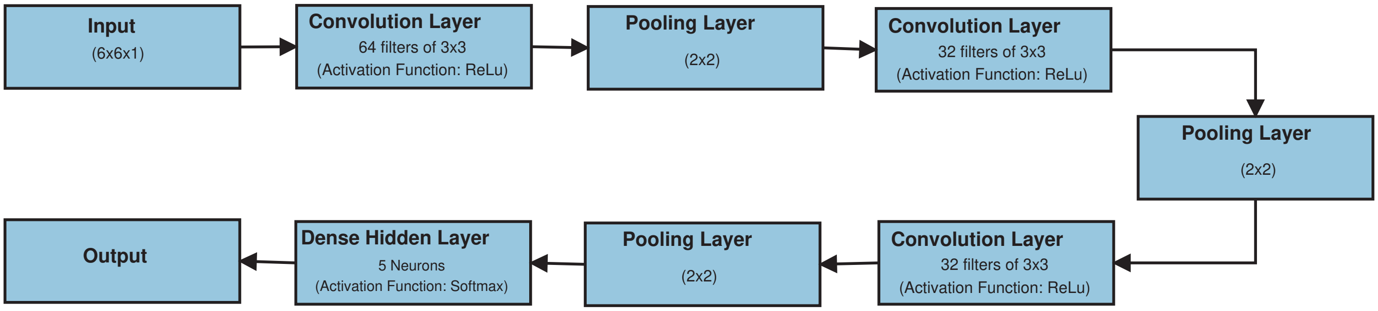

Fig. 3. Flowchart of CNN model

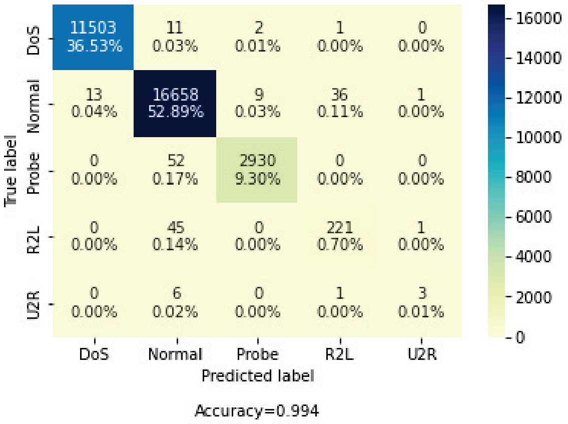

Fig. 4. Confusion Matrix of 2D-CNN

- Accuracy (AC) is the fraction of total samples correctly classified as X or Y.
- Precision (P) is the fraction of samples correctly classified as X to the total number of samples classified as X.

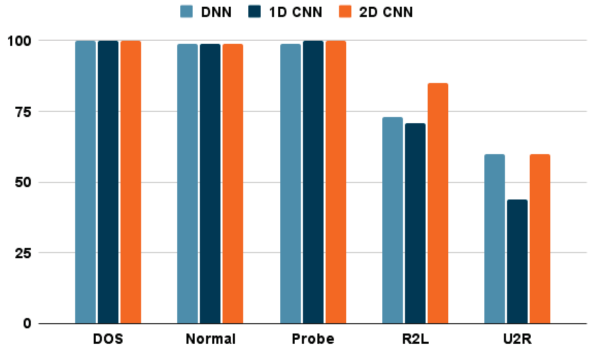

Fig. 5. Precision of DNN, 1D-CNN and 2D-CNN models

- Recall(R) is the fraction of samples correctly classified as X to the actual number of X samples in the dataset.
- F-measure (F1-score) is the harmonic mean of precision and recall.

Our CNN model with a 0.001 learning rate, 20 epochs, 3x3 kernel size, and pooling size of 2x2 achieved the

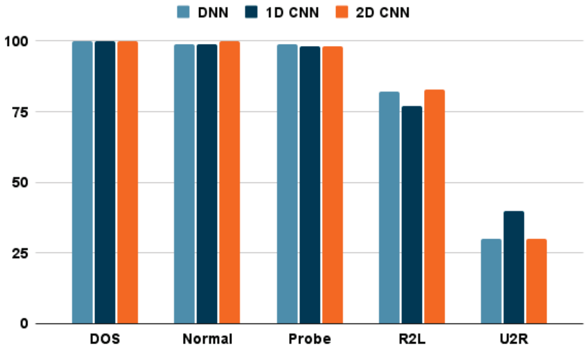

Fig. 6. Recall of DNN, 1D-CNN and 2D-CNN models

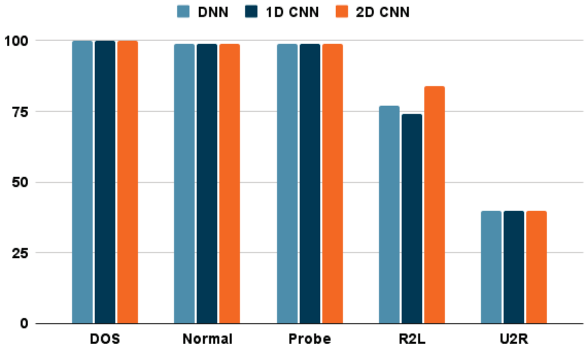

Fig. 7. F1-Score of DNN, 1D-CNN and 2D-CNN models

training and testing accuracy of 0.994. The confusion matrix of the model is shown in fig 4, which provides the matrix of prediction and actual class samples for test data in classification. CNN model achieved higher accuracy with high precision, recall, and F1-score as shown in Fig. 5, Fig. 6, and 7. However, the training time of the DNN model is less as compared to the CNN model, as shown in Fig. 8. CNN model achieved the highest accuracy with reduced loss, as shown in Fig. 9 and Fig.10.

CONCLUSION

Researchers are using different deep learning (DL) techniques for detecting intruders in IoT networks. In this paper, we proposed a 2D-CNN model for NIDS and showed that the CNN model could be used for tabular data. The accuracy of three models DNN, 1D-CNN, 2D-CNN models, for intrusion detection is measured. We found that the accuracy of the 2D-CNN model is highest with high precision, recall, and F1-Score. Our future work deals with the class imbalance problem and then compares the accuracy. Reduce the runtime of the CNN model and tune the model with different techniques. Real-time detection is needed in IoT systems, so the model should detect the new unknown attacks.

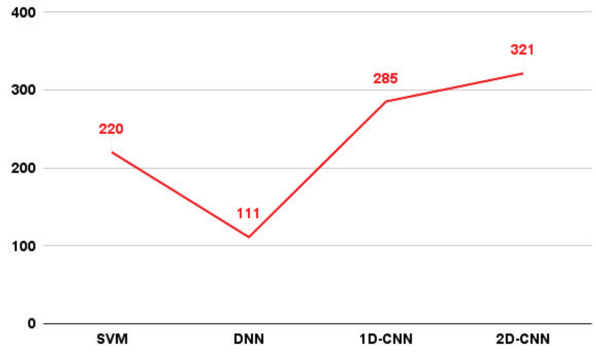

Fig. 8. Training Time of different models

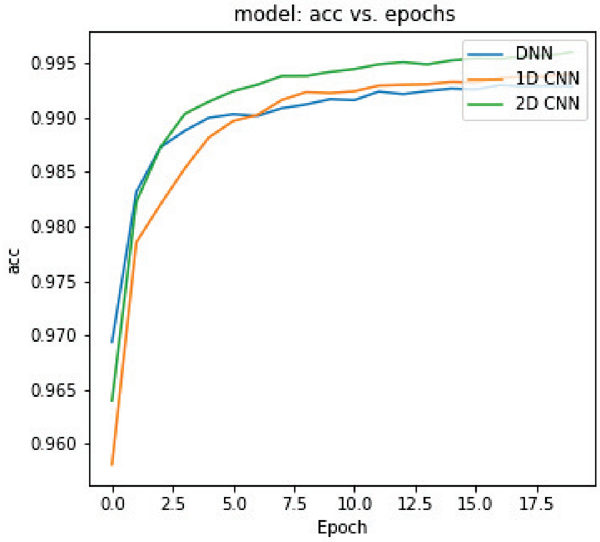

Fig. 9. Accuracy vs. Epochs of DNN, 1D-CNN and 2D-CNN models

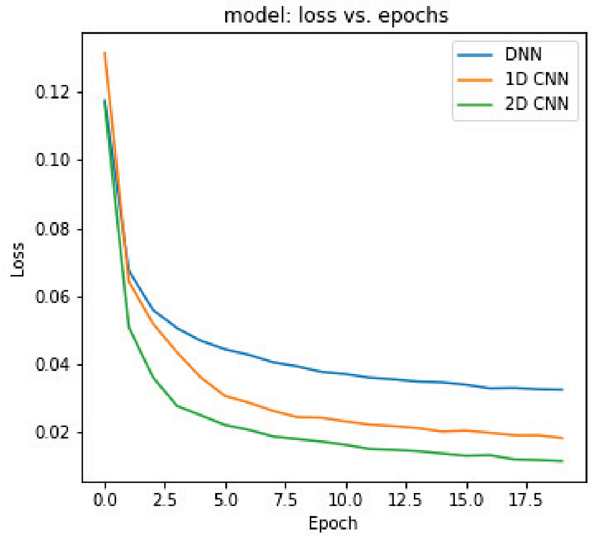

Fig. 10. Loss vs. Epochs of DNN, 1D-CNN and 2D-CNN models

REFERENCES

[1] Xu, L. Da, Member, S., He, W., & Li, S. (2014). Internet of Things in Industries: A Survey. IEEE Transctions on Industrial Informatics 10(4), 2233–2243.

[2] Al-fuqaha, A., Member, S., Guizani, M., Mohammadi, M., & Member, S. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications, IEEE Communication surveys & 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 of Things Journal, 4(5), 1125–1142.

[4] Chaabouni, N., Mosbah, M., Zemmari, A., Sauvignac, C., & Faruki, P. (2019). Network Intrusion Detection for IoT Security Based on Learning Techniques. IEEE Communications Surveys and Tutorials, 21(3), 2671–2701.

[5] 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. Transactions on Emerging Telecommunications Technologies, 32(1), 1–29.

[6] Al-garadi, M. A., Mohamed, A., Al-ali, A., Du, X., Guizani, M. (2020). A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. 1–42.

[7] N. Shone, T. N. Ngc, V. D. Phai, and Q. Shi, ”A deep learning approach to network intrusion detection,” IEEE Transactions on Emerging Topics in Computational Intelligence, vol. 2, pp. 41–50, 2018.

[8] M. Al-Zewairi, S. Almajali, and A. Awajan, ”Experimental Evaluation of a Multi-layer Feed-Forward Artificial Neural Network Classifier for Network Intrusion Detection System,” in Proceedings of the 2017 International Conference on New Trends in Computing Sciences (ICTCS), 2017, pp. 167–172.

[9] Alrashdi, I., & Alqazzaz, A. (2019). AD-IoT : Anomaly Detection of IoT Cyberattacks 1n Smart City Using Machine Leaming. 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), 305– 310.

[10] 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.

[11] A. K. Verma, P. Kaushik and G. Shrivastava, ”A Network Intrusion Detection Approach Using Variant of Convolution Neural Network,” 2019 International Conference on Communication and Electronics Systems (ICCES), 2019, pp. 409-416, doi: 10.1109/ICCES45898.2019.9002221.

[12] Ge, M., Syed, N. F., Fu, X., Baig, Z., and Robles-Kelly, A. (2021). Towards a deep learning-driven intrusion detection approach for Internet of Things. Computer Networks, 186(December 2020), 107784. https://doi.org/10.1016/j.comnet.2020.107784.

[13] 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. Algorithms for Intelligent Systems. Springer, Singapore.

[14] Fenanir, S., Semchedine, F., & Baadache, A. (2019). Revue d' Intelligence Artificielle A Machine Learning-Based Lightweight Intrusion Detection System for the Internet of Things. 33(3), 203–211.

[15] B. Sharma, L. Sharma and C. Lal, ”Anomaly Detection Techniques using Deep Learning in IoT: A Survey,” 2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE), Dubai, United Arab Emirates, 2019, pp. 146-149

[16] ”NSL-KDD — Datasets — Research — Canadian Institute for Cybersecurity — UNB.” [Online]. Available: https://www.unb.ca/cic/datasets/nsl.html.