The Role of CNN for Intrusion Detection Systems: An Improved CNN Learning Approach for SDNs

Mahmoud Said Elsayed[1], Hamed Z. Jahromi[1], Muhammad Mohsin Nazir[2], and Anca Delia Jurcut[1(✉)]

[1] University College Dublin, Dublin, Ireland
{mahmoud.abdallah,hamed.jahromi}@ucdconnect.ie, anca.jurcut@ucd.ie
[2] Lahore College for Women University, Lahore, Pakistan
mohsin.nazir@lcwu.edu.pk

Abstract. An intrusion detection system (IDS) is an essential component of computer networks to detect and secure the system and environment from malicious activities and anomalous attacks. The convolutional neural network (CNN) is a popular deep learning algorithm that has been broadly applied in the field of computer vision. More recently, several researchers attempted to apply CNN for IDSs. However, the majority of these ignore the influence of the overfitting problem with the implementation of deep learning algorithms, which can impact the robustness of CNN-based anomaly detection systems. In this paper, we investigate the use of CNN for IDSs and propose a technique to enhance its performance by using two popular regularization techniques to address the overfitting problem. Our technique improves the capability of IDSs in detection of unseen intrusion events. We use InSDN benchmark dataset to train and evaluate the performance of our technique. The experimental results demonstrate that the regularization methods can improve the performance of CNN-based anomaly detection models for the software-defined networking (SDN) environment.

Keywords: Intrusion · Intrusion Detection System (IDS) · Machine learning · Software-defined Networking (SDN) · Convolutional Neural Network (CNN) · Overffiting problem

1 Introduction

Intrusion Detection Systems (IDSs) detect network intrusions (i.e., anomalies) by inspecting the network traffic flows to classify them as normal or anomalous. IDS employs preventive measures for anomalous activities and protect the environment from possible disruptions.

IDS commonly performs classification based on heuristics or rules. The system learns about the normal network activities and classify a particular traffic

D. Perakovic and L. Knapcikova (Eds.): FABULOUS 2021, LNICST 382, pp. 91–104, 2021.
https://doi.org/10.1007/978-3-030-78459-1_7

flow as malicious when it falls out of the normal conditions. However, it is challenging to distinguish and classify the attacks from the normal network traffic with regular traffic characteristics. An intruder may generate several network attacks similar to the normal traffic without being detected.

Researchers have been working actively to improve the IDS capabilities in detecting and classifying anomalous activities. However, reducing the false alarm rate and improving the detection rate is still an open research area.

More recently, machine learning techniques have been employed in IDS and proved to be effective for detecting anomalous activities [1–5]. For instance, algorithms such as Logistic Regression (LR), Naïve Bayes (NB), and Support Vector Machine (SVM) are widely used to identify various network-based attacks. These aforementioned techniques are known as shallow learning due to its learning characteristics based on pre-defined features. These approaches uses features engineering as domain knowledge to extract features using data mining techniques. However, the feature selection is a challenging process and requires the domain of knowledge experts. For example, a feature that may exist is an important characteristic for one attack class, but irrelevant to other types of attacks [6]. Besides, the nature of data takes the behavior of non-linearity, and the shallow learning methods provide low accuracy when the data account the high degree of non-linearity. As a result, the researchers have started to use deep learning techniques for attack recognition tasks [7–12].

Deep learning (DL) techniques can automatically extract the high-level features from the network traffic without using feature engineering or knowledge of domain experts. The CNN is broadly applied for various classification and recognition tasks in computer vision and other research domains (e.g., identifying key chemical structures in various drug developments, weather phenomena and video surveillance). The strength of CNN as compared to the others DL models lies in it's ability to extract high-level features from raw data with low number of parameters(i.e., the concept of weight sharing). Furthermore, the CNN significantly outperforms the traditional feature selection algorithms in obtaining the best representative features.

The success of CNN in various domains is encouraging for the network researchers to investigate it's effectiveness for the intrusion detection tasks. The features learning capability of CNN makes it suitable for today's explosive network environment. However, the overfitting problem of CNN has been hitherto neglected in the majority of intrusion detection studies. The performance of the neural networks (NN) relies on the input data and a set of parameters. Therefore, to produce an expressive model with a capability of learning complicated relationships between inputs and outputs features, a complex network architecture with multi-hidden layers can be used. The complex network architecture, however, requires a large amount of training data, which is not always available. Consequently, training large neural networks with limited data can cause the model to describe the random error in the data rather than the relationships between variables. This problem leads to model overfitting issues. This model

can perform well during the training but perform poorly in a real test data (even if they come from the same distribution).

Several methods exist to address the overfitting problem, and one of the effective methods uses the regularization techniques. The key idea behind the regularization is to impose a large number of restrictions on the learning model or simply control the model's ability to learn. Although it sacrifices some flexibility to fit the training data effectively, it avoids the model to become complex. Hence, it's ability to predict unseen data can be increased efficiently (i.e., anomaly detection). In this paper, we investigate the use of CNN for IDSs, and propose a technique to enhance it's performance using two popular regularization techniques i.e., L2 regularization and the dropout methods.

2 Background

This section briefly introduces the background related to the proposed work. The first subsection presents the current challenges of the conventional networks, the motivation for the SDN to overcome such limitations and the current security flaws of SDN. The second subsection introduces the component of the CNN and it's working operation. The last subsection discusses the overfitting problem, the theoretical background and the existing techniques to address similar kind of problems.

2.1 Software-defined Networking (SDN) Environment

In the conventional distributed networks, the routers and the switches operate independently. Each device makes an independent decision on how to handle or treat the network traffic. Consequently, a change in the network policy or routing configuration requires access to each device individually. The aforementioned caveat of the distributed networks is due to the existence of control plane (responsible for decision making) and the data plane (responsible for forwarding packets/frames from one interface to another) within a same network device. Thus, the network management and the orchestration is often complex and relatively exposed to errors. For example, in a large scale environment with various network equipment and connectivities, it is a complex process to develop innovative services without jeopardizing the network stability.

The emergence of SDN and programmable networks with centralized architecture facilitates network research studies to explore, develop and implement a new level of network security and management applications [13,14]. A SDN environment offers full control and visibility over the network, and facilitates the collection of end-to-end network heuristics from a centralized point [15]. SDN controllers interact with the network devices using southbound interface (SBI), and provide network application programming interfaces (APIs) to the applications using northbound interface (NBI).

SDN allows the developers to create network applications and implement it directly through API, avoiding the high cost of hardware devices. For instance,

an IPS/IDS system can utilize SDN's NBI to advise the controller on how to deal with a malicious traffic flows. Despite the advantage of SDN environments, the centralized architecture can produce new attack vectors that are specific to SDNs. For example, SDN controller is the brain of the network, and is a single point of failure. If an attacker exhaust the controller resources by sending a massive amount of malicious traffic (e.g., Denial of Service (DoS) or Distributed Denial of Service (DDoS)), the network operation might be impacted and cause a significant damage. Similarly, a compromised SDN controller gives the attacker a chance to modify or redirect the network traffic [16].

2.2 Convolutional Neural Network (CNN)

CNNs are organized in different architectures compared to the traditional neural networks. We can find that each layer in the traditional neural network consists of a group of neurons, which are connected to all neurons in the previous layer. In contrast, in CNN, each layer is attached only to a small portion of the neurons instead of fully connected neurons with the previous layer. A typical structure of the CNN is constructed from three layers of convolution, pooling, and fully connected layer [17]. In convolutional layer, a filter or kernel passes through the input image and forms a conclusion of an array of numbers. Multiplying the kernel across the portion of the input produces a single number, by moving the filter through the whole image. A metric of multiple numbers is generated, which refers to a feature map. Using multiple kernels produces a number of feature maps, which represent different characteristics of the input tensors. The mathematical description of the convolution layer is described in the following equation:

$$M_i = f(M_{i-1} \otimes W_i + b_i) \tag{1}$$

Where M_i describes the feature map at layer i and $M_0 = \mathrm{X}$ (input layer), W_i represents the weight vector of the convolution filter at layer i, while the b_i and f represent the bias vector and activation function, respectively. One of the typical non-linear function used in the CNN is rectified linear unit (ReLU) activation function [18]. Compared to the regular neural network, the dominant CNN is using less number of parameters by sharing the same weight and bias vector. It also does not require hand-crafted feature extraction as compared to conventional machine learning classifiers. The second layer is the pooling layer i.e., downsampling operation, and aims to reduce the dimensionality of the feature map.

There are two types of pooling operation: Max pooling and average pooling [19]. The final convolution or polling layer's output is passed through one or more fully-connected layers for classification tasks to get the final outputs. The number of nodes or neurons in the last output layer is equal to the number of output classes.

2.3 Overfitting Prevention and Regularization

Overfitting is a serious problem in machine learning techniques due to high number of parameters. With the occurrence of overfitting, learning model performs well during the training phase, but provides a poor result on unseen data. A predominant solution to deal with overfitting problem is by applying some model selection algorithm to choose the features with greater importance. However, these techniques may cause some loss of valuable information. An alternative solution is using regularization methods to control the model complexity to reduce the burden on the parameters complexity (i.e. weights & biases). The regularization techniques penalize the features by imposing a constraint on the magnitude of the coefficients without any loss. Controlling the value of parameters can eliminate overfitting and enhance the performance of the model on unseen data. The L2 regularization is a method which adds a penalty to the square of the weight coefficient values. As a result, it poses the large parameters be close to zero. During the training process, we try to minimize the following cost function:

$$j(w^1, b^1, ..., w^L, b^L) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)}) \tag{2}$$

where, L is the loss function, w is the wight and b is the bias. Now, using L2 regularization, the loss function will become:

$$j(w^1, b^1, ..., w^L, b^L) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)}) + \frac{\lambda}{2m} \sum_{l=1}^{L} \left\| w^L \right\|^2 \tag{3}$$

where, λ is a parameter that can be tuned to control the regularization effect. Using large λ, the weight penalty will be large. Similarly, small λ will reduce the effect of regularization. This is trivial, because the cost function must be minimized. By adding the squared norm of the weight matrix and multiplying it by λ, large weights will be driven down in order to minimize the cost function.

Hinton et al. proposed dropout method to reduce the risk of the overfitting [20]. The key idea behind dropout is to randomly ignore some units from the network with probability p during training, but use all these units in testing. The best value of the probability is usually between 0.5 and 0.8.

An example of the dropout process is illustrated in Fig. 1 and Fig. 2. In this work, we combine L2 regularization (i.e. weight decay) and dropout techniques together to improve the performance of our proposed detection model.

3 Related Work

Khan et al. applied CNN for intrusion detection on the KDD'99 dataset [21]. The model architecture composed of three hidden layer. Each hidden layer contains a convolutional layer followed by a pooling layer. They demonstrated that their proposed CNN model outperforms support vector machine (SVM) and deep

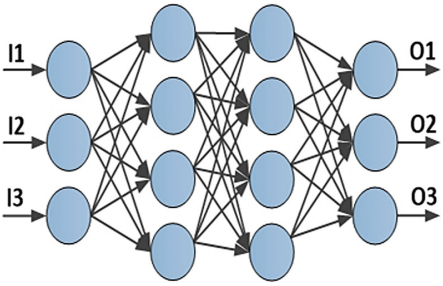

Fig. 1. A typical feedforward ANN.

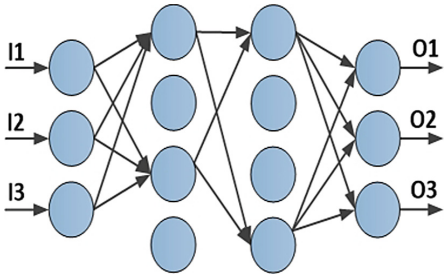

Fig. 2. The dropout of p = 0.5 applied.

belief network (DBN) with an accuracy of 99.23%. The estimated accuracy for SVM and DBN are 98.20% and 98.59%, respectively.

Yong et al. used batch normalization to improve the performance of the CNN model for intrusion detection on KDD' 99 dataset [22]. The best model performance was obtained using two convolutional layers with kernel dimensions of 12, and 24, respectively. Their experimental results reveals that CNN is more effective for intrusion detection compared to SVM and neural network algorithms.

HU et al. improved convolutional neural network (CNN) by using adaptive synthetic sampling (ADASYN) algorithm [23]. The ADASYN aims to deal with unbalanced data by adding new samples for minor classes and prevent the model from being biased towards frequent samples. In addition, to improve the diversity of features, the split convolution module SPC-CNN method is used to overcome the problem of inter-channels information redundancy. The enhanced CNN model (SPC-CNN) achieved an accuracy of 83.83% on the NSL-KDD testing data, with 4.35% higher than the traditional CNN.

XIAO et al. proposed an intrusion detection approach based on CNN model using KDDcup99 dataset [24]. The architecture of the CNN based method was constructed from two convolutional layers, two Max pooling and one fully connected layer. They have used the principal component analysis (PCA) and the autoencoder reduction techniques to reduce the feature of the input data and to prepare the dimensional input for the CNN requirements. Two sets of feature reduction, 100 and 121 are obtained using PCA, while different sets of 100, 121, 81, and 64 are extracted from the autoencoder. From their experimental results it can be understood that the autoencoders have been more effective in feature reduction, in comparison to the PCA technique.

JIANG et al. proposed a hybrid intrusion detection model (CNN-BiLSTM). The model integrates the CNN and the Bi-directional long short-term memory (BiLSTM), in order to learn the spatial and temporal features [25]. The synthetic minority oversampling technique (SMOTE) algorithm is used to tackle the problem of minority classes in the unbalanced data. Similarly, the one-side selection (OSS) algorithm is used to reduce the major class samples. Two datasets NSL-KDD and UNSW-NB15 are used to evaluate their proposed model.

The CNN-BiLSTM model achieved an overall accuracy of 82.74% and 77.16% for NSL-KDD and UNSW-NB15, respectively.

4 Proposed CNN-Based Technique

This section introduces the solution methodology of the proposed CNN model. The first subsection introduces the dataset, which is used to verify the proposed detection model. The second subsection shows the various steps to prepare the dataset for the model training. In the last subsection, we discuss the structure of the CNN model and the used parameters to set up our proposed method.

4.1 Dataset

In this work, we use InSDN dataset to evaluate the performance of the proposed CNN approach [6]. InSDN dataset allows researchers to investigate and develop intrusion detection models specific to SDN networks. The dataset includes normal traffic (e.g., HTTP, HTTPS, DNS, Email, FTP and SSH), attacks related to traditional networks and SDN related attacks. The attack types include DoS, DDoS, Probe, Botnet, Exploitation, Password-Guessing, and Web attacks. The dataset was produced using a virtual SDN network, which was constructed from four virtual machines. The first machine acts as an attacker and the second machine is the Metasploitable-2 vulnerable Linux server. The other two machines are used to represent the OVS switch and SDN controller. In addition, the dataset's attacks come from several sources, i.e., internal and external, to reflect the real attack cases.

The dataset is available in both PCAP file and .CSV file format and divided into three groups. The first group i.e., OVS group, includes the attacks coming from outside to SDN internal network. The second group contains the attacks against Metasploitable 2 server. The last group represents the normal traffic. The dataset has more than 80 features generated using the CICFlowMeter tool. In this work, we only utilize a subset of 48 features to train our CNN model. More details about these features are discussed in [6].

Furthermore, the attack samples, which were used during the testing phase are from different distribution from those used in the training. The number of data records for training and testing datasets is 135,870 and 50,230, respectively.

4.2 Data Preprocessing

The following pre-processing steps are performed before feeding the input data into the deep learning model:

1. The normalization technique is applied to map the value of features between 0 and 1, using the standardization method (i.e., Z-score normalization) according to the Eq. 4.
2. The 1D-dimensional network traffic is translated into image format with a dimension of 8×6.
3. the Symbolic feature is converted into numerical data.

The label column has several attack classes and a normal class. It is also important to note that this study preforms a binary classification and set the normal label to 0 and all attack labels to 1.

$$y = \frac{y - MIN}{MAX - MIN} \tag{4}$$

4.3 Proposed CNN Architecture

The proposed CNN architecture is depicted in Fig. 3. The network architecture is constructed from two convolutional layers with 32, 64 filters, where each filter has a size of 3×3 and a stride of 1. Each convolutional layer is followed by a max-pooling layer with a size of 2×2. The 'same' padding is used in the convolutional layers to make the output same as the input. Thus, the input image gets fully covered by the filter and the specified stride. A fully connected layer is used with a number of units equal to 128. The final layer is the softmax layer to classify the network traffic into normal (0) or malicious traffic (1).

The binary cross-entropy is used as a loss function in the output layer, while the non-linear ReLU activation function is used for all layers except the last layer. The drouput method is used after the second Max pooling layer and the fully connected layer. The probability (p) of 0.2 is used in prior after the softmax layer, while $p = 0.5$ is used after the fully layer.

Since the dropout causes some information loss, hence any loss in the first layers will propagate to the whole network. Therefore, a common practice is to start with low dropout and then gradually increase it. In addition to the dropout layer, the weight decay technique is used to further improve the performance of the model, and to reduce the error rate. L2 regularization method is applied in this experiments, as it provides high performance than the L1 method.

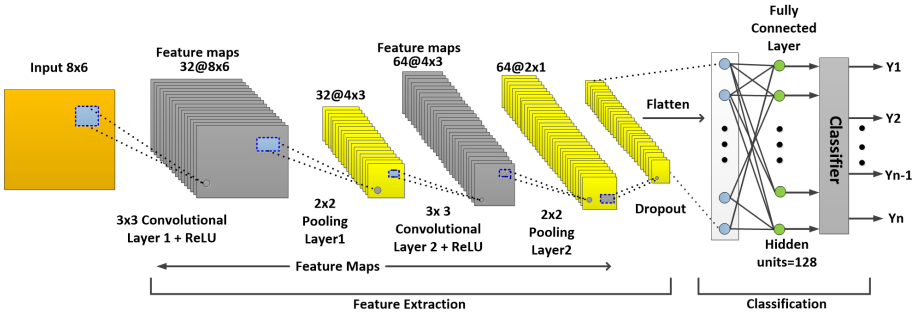

Fig. 3. The structure of CNN model.

5 Evaluation

This section discuss the experimental results of the proposed model.

5.1 Evaluation Criteria

To evaluate our CNN model, four performance indicators of Accuracy (Acc), Precision (Pre), Recall (Rec) and F1 measure are used. The mathematical representation of these indicators are calculated based on Eqs. 5, 6, 7, and 8, respectively.

$$\text{Acc} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$\text{Pre} = \frac{TP}{TP + FP} \tag{6}$$

$$\text{Rec} = \frac{TP}{TP + FN} \tag{7}$$

$$\text{F1} = \frac{2 \times \text{Pre} \times \text{Rec}}{\text{Pre} + \text{Rec}} \tag{8}$$

TP, TN, FP and FN denote true positives, true negatives, false positives, and false negatives, respectively.

5.2 Experimental Results

We use Keras Library with Python programming language to perform all our experiments. The experiments are conducted on a machine with Intel core I7-8650U CPU @ 1.90 GHz (8 cores), Window 10 operating system. The InSDN is the benchmark dataset used for all classification experiments to perform 2-class anomaly detection.

We compare the performance of the CNN model with various classification algorithms, including k-nearest neighbor classifier (KNN), naive Bayes (NB), support vector machines (SVM), Adaptive Boosting (AdaBoost), logistic regression (LR), random forest (RF), decision tree (DT). Besides the aforementioned classifiers, we use deep neural network (DNN) to further evaluate our CNN model. The used parameters of the DNN is depicted in Fig. 1, while the default parameters from the scikit-learn are used for all other classifiers.

The experiment results are represented in Table 2, Fig. 4 and Fig. 5. It can be seen that the proposed CNN algorithm performs well compared to all other existing classifications. Table 2 and Fig. 4 show that the classical ML algorithms have low performance compared to the deep learning models, except the KNN algorithm, with its performance slightly higher than the DNN model. The average accuracy of the KNN and DNN is 89.06% and 88.32%, respectively. The NB algorithm performed very poor in all evaluation metrics followed by RF

and AdaBoost. On the other side, we can see that the CNN model generally outperforms the classical ML algorithms.

In addition, the classification of normal class in all models is relativity lower than the attack class. This is due to the low size of the normal class during training and testing phases. The low amount of samples is not enough for the classifier to learn the behavior of raw data significantly. To further evaluate our proposed CNN based model, the receiver operating characteristics (ROC) curve (Fig. 5) is used to represent how the model performs in general. The ROC curve represents the relation between the true-positive and false-positive rates, and the area under the curve (AUC), as a measure of the model's capability. The CNN has a higher AUC with a value of 0.928, followed by the KNN and DNN algorithms with values of 0.89 and 0.88, respectively. In contrast, all other classifiers provide low AUC values, which indicate a low performance for these algorithms in network anomaly detection (Table 1).

Table 1. The hyper-parameters used in multi-layer perceptron approach.

Traffic type	Detection rate
Hidden nodes (HN)	3
Number of neurons	100, 100, 100
Number of epoch	10
Batch size	128
Classification function	Softmax
Activation function	ReLU

Table 2. Precision, Recall, and F1-score of the different methods.

Model	Precision (%)		Recall (%)		F1-Score (%)	
	Normal	Attack	Normal	Attack	Normal	Attack
NB	50.21	63.87	90.30	16.05	64.53	25.66
RF	52.02	75.00	93.03	19.58	66.73	31.05
AdaBoost	52.68	65.08	82.53	30.51	64.31	41.54
DT	81.36	59.91	33.78	92.74	47.74	72.79
LR	96.89	59.95	29.36	99.11	45.07	74.71
SVM	81.34	75.58	70.78	8478	75.69	79.92
DNN	96.47	83.00	78.73	97.30	86.71	89.58
KNN	89.62	88.56	87.52	90.50	88.56	89.52
Proposed CNN Model	**98.67**	**88.82**	**86.71**	**98.91**	**92.31**	**93.59**

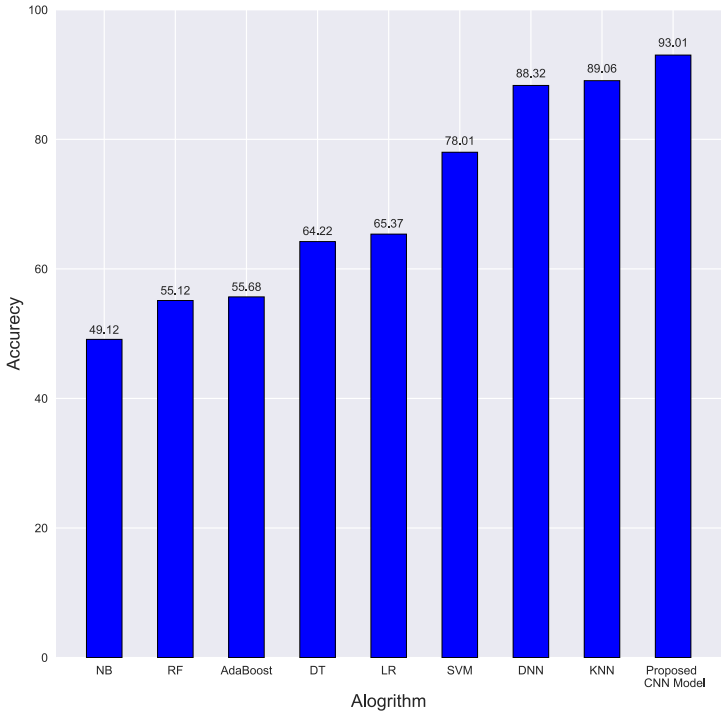

Fig. 4. Comparison of the CNN model and other algorithms

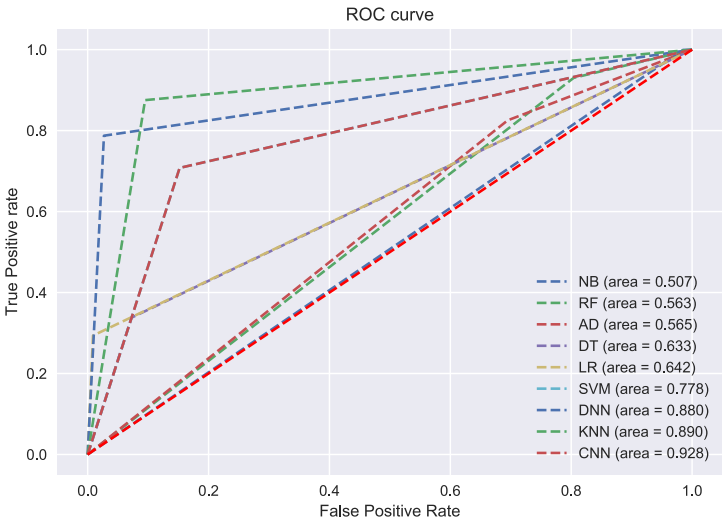

Fig. 5. Receiver operating curve (ROC) of different IDS approaches.

6 Discussion

The CNN has received significant attention in various applications such as natural language processing and image recognition. It has higher performance in discovering the high-dimensions of the input data [26,27]. The convolution layers are using a set of filters or kernels and build them on top of each other to learn the complex features. Each layer can learn some specific features of the input data. For example, the first layers detect edges, the next layers combine them to detect shapes, while the following layers merge this information to detect that as a feature. The CNN uses the minimum number of training parameters (i.e., weight sharing), which is a significant advantage compared to other deep learning models. It can reduce the time used for the training process and also able to classify the network traffic very fast. The CNN can also be used comprehensively for network environments since the network packets are similar to the structure of the words, and the traffic is similar to sentences and articles. However, the adoption of the CNN for the anomaly detection area is still at early stage and needs further research.

One of the main challenges, which seriously hinder the performance of the ML/DL models is the problem of Overfitting. The model can effectively perform very well during the training but fails to display a good tendency with the unseen data. There are many reasons that can cause this problem such as, the complexity of the model and the low amount of data used to create a suitable approach. However, most of machine learning algorithms are data-hungry and collecting a large amount of training data can be considered one of the significant challenges, especially in the network intrusion domain. The availability of network data or creating a new dataset can be subjected to several challenges including privacy or illegal issues. The network data includes customer information or sensitive data, and the availability of such data can reveal this information to the public. Adding to the previous problem, many existing works in the intrusion detection area, used the same distribution of testing data similar to those used during the training. Evaluating the proposed models using such methods is not a reliable method for anomaly detection, since any simple algorithm can give very high accuracy that can reach to 99% or higher. However, employing the same model for new data (i.e. zero-day attacks) will cause very high false rates and low performance as well.

Thus, the best practice to test the efficacy of intrusion detection models is to evaluate how it can work with new data that have been never seen before and during the training. This is what we investigated and successfully achieved in this work. Although the obtained accuracy (93.01%) of our proposed CNN based technique is not convincing enough to be applied in real SDN networks, we are working to improve the performance of the proposed CNN model by finding new methods to solve the problem of the Overfitting, and hence, increasing the ability of the model for the outlier detection.

7 Conclusion

The innovative features of the SDN speed up replacing the current network architectures, especially in the recent days since cyber-attacks are becoming more prevalent. With a growing number of people working remotely, the traditional network architecture is starting to fall short of satisfying the essential network needs. On the other hand, the SDN paradigm generates security risks that still need to be addressed before securely deploying it in the current networks. In this work, we investigated the role of the CNN to tackle the security issues of the SDN by proposing an improved CNN-based IDS technique. The dropout and L2 regularization techniques are used to overcome the problem of overfitting and to enhance the technique functionality to detect unseen anomalies. The experimental results indicated that the CNN could effectively be used to detect the new anomalies compared to other classifiers. In our future work, we will work to improve the performance of the CNN for IDS by proposing further techniques to enhance its ability for anomaly detection.

References

1. Çavuşoğlu, Ü.: A new hybrid approach for intrusion detection using machine learning methods. Appl. Intell. **49**(7), 2735–2761 (2019). https://doi.org/10.1007/s10489-018-01408-x
2. Halimaa, A., Sundarakantham, K.: Machine learning based intrusion detection system. In: 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), pp. 916–920. IEEE (2019)
3. Abdulhammed, R., Musafer, H., Alessa, A., Faezipour, M., Abuzneid, A.: Features dimensionality reduction approaches for machine learning based network intrusion detection. Electronics **8**(3), 322 (2019)
4. Alkasassbeh, M., Almseidin, M.: Machine learning methods for network intrusion detection. arXiv preprint arXiv:1809.02610 (2018)
5. Taher, K.A., Jisan, B.M.Y., Rahman, M.M.: Network intrusion detection using supervised machine learning technique with feature selection. In: 2019 International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), pp. 643–646. IEEE (2019)
6. Elsayed, M.S., Le-Khac, N.-A., Jurcut, A.D.: InSDN: a novel SDN intrusion dataset. IEEE Access **8**, 165 263–165 284 (2020)
7. Althubiti, S.A., Jones, E.M., Roy, K.: LSTM for anomaly-based network intrusion detection. In: 2018 28th International Telecommunication Networks and Applications Conference (ITNAC), pp. 1–3. IEEE (2018)
8. Elsayed, M.S., Le-Khac, N.-A., Dev, S., Jurcut, A.D.: DDoSNet: a deep-learning model for detecting network attacks. In: 2020 IEEE 21st International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), pp. 391–396. IEEE (2020)
9. Elsayed, M.S., Le-Khac, N.-A., Jurcut, A.D.: Detecting abnormal traffic in large-scale networks. In: 2020 International Symposium on Networks, Computers and Communications (ISNCC), pp. 1–7. IEEE (2020)
10. Said Elsayed, M., Le-Khac, N.-A., Dev, S., Jurcut, A.D.: Network anomaly detection using LSTM based autoencoder. In: Proceedings of the 16th ACM Symposium on QoS and Security for Wireless and Mobile Networks, pp. 37–45 (2020)

11. Shone, N., Ngoc, T.N., Phai, V.D., Shi, Q.: A deep learning approach to network intrusion detection. IEEE Trans. Emerg. Top. Comput. Intell. **2**(1), 41–50 (2018)
12. Al-Qatf, M., Lasheng, Y., Al-Habib, M., Al-Sabahi, K.: Deep learning approach combining sparse autoencoder with SVM for network intrusion detection. IEEE Access **6**, 52 843–52 856 (2018)
13. Elsayed, M.S., Le-Khac, N.-A., Jurcut, A.D.: Dealing with covid-19 network traffic spikes [cybercrime and forensics]. IEEE Secur. Priv. **19**(1), 90–94 (2021)
14. Jahromi, H.Z., Hines, A., Delaney, D.T.: Towards application-aware networking: ML-based end-to-end application KPI/QoE metrics characterization in SDN. In: 2018 Tenth International Conference on Ubiquitous and Future Networks (ICUFN), pp. 126–131. IEEE (2018)
15. Jahromi, H.Z., Delaney, D.T.: An application awareness framework based on SDN and machine learning: defining the roadmap and challenges. In: 2018 10th International Conference on Communication Software and Networks (ICCSN), pp. 411–416. IEEE (2018)
16. Scott-Hayward, S., O'Callaghan, G., Sezer, S.: SDN security: a survey. In: 2013 IEEE SDN for Future Networks and Services (SDN4FNS), pp. 1–7. IEEE (2013)
17. Vedaldi, A., Lenc, K.: MatConvNet: convolutional neural networks for MATLAB. In: Proceedings of the 23rd ACM International Conference on Multimedia, pp. 689–692 (2015)
18. Zhou, D., Yan, Z., Fu, Y., Yao, Z.: A survey on network data collection. J. Netw. Comput. Appl. **116**, 9–23 (2018)
19. Yamashita, R., Nishio, M., Do, R.K.G., Togashi, K.: Convolutional neural networks: an overview and application in radiology. Insights Imaging **9**(4), 611–629 (2018). https://doi.org/10.1007/s13244-018-0639-9
20. Hinton, G.E., Osindero, S., Teh, Y.-W.: A fast learning algorithm for deep belief nets. Neural Comput. **18**(7), 1527–1554 (2006)
21. Khan, R.U., Zhang, X., Alazab, M., Kumar, R.: An improved convolutional neural network model for intrusion detection in networks. In: 2019 Cybersecurity and Cyberforensics Conference (CCC), pp. 74–77. IEEE (2019)
22. Yong, L., Bo, Z.: An intrusion detection model based on multi-scale CNN. In: 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), pp. 214–218. IEEE (2019)
23. Hu, Z., Wang, L., Qi, L., Li, Y., Yang, W.: A novel wireless network intrusion detection method based on adaptive synthetic sampling and an improved convolutional neural network. IEEE Access **8**, 195 741–195 751 (2020)
24. Xiao, Y., Xing, C., Zhang, T., Zhao, Z.: An intrusion detection model based on feature reduction and convolutional neural networks. IEEE Access **7**, 42 210–42 219 (2019)
25. Jiang, K., Wang, W., Wang, A., Wu, H.: Network intrusion detection combined hybrid sampling with deep hierarchical network. IEEE Access **8**, 32 464–32 476 (2020)
26. Gu, J., et al.: Recent advances in convolutional neural networks. Pattern Recogn. **77**, 354–377 (2018)
27. Khan, A., Sohail, A., Zahoora, U., Qureshi, A.S.: A survey of the recent architectures of deep convolutional neural networks. Artif. Intell. Rev. **53**(8), 5455–5516 (2020). https://doi.org/10.1007/s10462-020-09825-6