Detecting Network Intrusions Using Convolutional Neural Networks

Nur Mahabub Morshedur Rahman
011 171 291
United International University
Dhaka, Bangladesh
nrahman171291@bscse.uiu.ac.bd

Mahbub-E-Sobhani
011 171 134
United International University
Dhaka, Bangladesh
msobhani171134@bscse.uiu.ac.bd

ABSTRACT

The world's dependency on the internet is ever-growing. Information is the most valuable resource in the world. To protect information from unwanted hands is of utmost importance. Information gets stolen when a network gets breached. An intrusion detection system finds known and unknown attacks that facilitate breaching of a network. In this paper, we model an intrusion detection system trained to detect such types of attack. We do so using neural networks which is a machine learning approach. Furthermore, we study the performance of our model in binary and multiclass classification environments. We look into the performance of other traditional machine learning algorithms such as Support Vector Machine, Naive Bayes, K-Nearest Neighbour and Random Forest on the data set (NSL-KDD) we use. Our experiments show that using Convolutional Neural Networks are better approaches to detect network intrusions.

1 INTRODUCTION

Society and the internet integrate deeply. This dependency is ever-growing, and for that enormous quantity of data is being produced. Security of information is a significant concern and to identify attacks on networks, especially attacks that are entirely new and unseen, is a necessary need. An intrusion detection system (IDS) can identify such attacks, whether ongoing or an attack that has already occurred. Signature-based systems are the traditional ways of detecting network intrusions, and they use signatures and patterns of attacks that have already occurred before. They are very accurate when detecting known security threats but are entirely ineffective in identifying unknown malicious threats. So, real-time threat detection scenarios do not use traditional detection systems. The self-learning system is a proactive method that uses machine learning concepts such as supervised and unsupervised algorithms to be on par with the ever-changing network attack techniques. Such a machine learning methods is Convolutional Neural Networks which have already been successful in fields like natural language processing (NLP) and computer vision (CV). Inspired by the performance of the method, we propose an intrusion detection system using it (CNN-IDS). The contributions of this paper are:

(1) We design and implement the CNN-IDS.
(2) We study the performance of our model in binary and multiclass classifications.
(3) We compare the performance of our models with that of Support Vector Machine, Naive Bayes, K-Nearest Neighbour and Random Forest on the data set we use (NSL-KDD) in binary and multiclass classification environments.

2 BACKGROUND

In this chapter we are presenting the primary studies in 2.1 along side with research papers that is aligned with our project work.

2.1 Preliminaries

This precursory segment introduces to the Grammatical Error Correction problem concepts and keywords that are required to have a full idea of understanding the rest of the sections without any difficulty.

Recurrent Neural Network

Recurrent Neural Network, also known as RNNs are a part of neural networks which is used for sequential prediction. RNNs allow previous outputs to be used as inputs and has the ability to process an input of any length by not increasing the model size where weights are shared across time.

Bidirectional Recurrent Neural Network

Bidirectional Recurrent Neural Networks is a variant of RNN. Which has all the advantages of basic RNN architecture with additional speed and capturing the ability of long-term dependencies. The ability of considering the future and past output for processing the current output made Bidirectional RNN so special for Natural Language Processing problems.

Long Short Term Memory

Long Short Term Memory, also known as LSTMs is a modified version of RNN having the ability of learning long-term dependencies. LSTM deals with the vanishing gradient problem encountered by traditional RNN. Remembering the information for long periods of time is the default behavior of LSTM. It has the advantage of forgetting something from the previous information when it needs to, and updates the necessary information for the future time steps.

Convolutional Neural Network

Convolutional Neural Network is most often used for image processing problems, but nowadays it has been discovered that CNN can also be useful for Natural Language Processing problems. Convolutional neural network is widely used for important contributions in deep learning and artificial intelligence. CNN is made up of multiple layers and has the fast processing ability because of powerful feature extraction and reduction techniques. The convolutional neural network is a feed-forward network that filters spatial data whereas recurrent neural networks feed data back to itself, and that is why it works well on sequential data.

3 RELATED WORK

In previous studies, traditional machine learning approaches such as Support Vector Machine, Naive Bayes, K-Nearest Neighbour and Random Forest have been successful in intrusion detection systems.[1] In recent years, a division of machine learning known as deep learning has been famous in this field due to its unparallel performance. Studies show that deep learning is better in terms of performance in detecting unknown network attacks than traditional methods.[2]

The authors of [1] have tested the effectiveness of Naive Bayes, Random Forest, Support vector machine and K-Nearest Neighbors on both encrypted and unencrypted network packets. With a dataset consisting of 1130 instances collected through packets captured by Wireshark, they have determined amongst all other algorithms, Random Forest is the algorithm which more accurate.

In [3], the researcher has experimented with all the methods of [1], but with a different dataset. They have used the popular KDD99 dataset, which is the precursor of the NSL-KDD dataset that we have used. With 21 types of attacks and 41 features in the dataset, they have used Information gain as their feature selection method. They have also concluded like [1] that Random Forest is more accurate than all the traditional machine learning algorithms.

Table 1: Result Summary of Paper [3]

Language	English
Algorithms	Naive Bayes, Random Forest, K-Nearest Neighbor, Decision Table
Dataset	KDD99
Methodology	1. Importing the dataset 2. Classifying and choosing the algorithm. 3. Using the 10-Fold method 4. Again testing using the Percentage Split (70%) 5. Checking the Correctly Classified Instance Percentage.
Evaluation Metrics	10-Fold cross validation, 70% percentage split
Results	10-Fold cross validation Naive Bayes: 76.16% Random Forest: 99.40% K-Nearest Neighbor: 98.94% Decision Table: 98.51%

[4] has applied CNN to the KDD99 dataset. They have found that a fixed learning rate of 0.1 performs the best. The paper claims that a deep learning-based approach such as CNN is suitable at modelling network traffic in comparison to other conventional machine learning classifiers. The authors of this paper say that the NSL-KDD dataset is outdated and have suggested other researches to use UNSW-NB15[5] dataset instead.

Table 2: Result Summary of Paper [4]

Language	English
Algorithms	CNN, CNN-LSTM, CNN-GRU, CNN-RNN
Dataset	KDD99
Methodology	Tensorflow > CNN 1 Layer > CNN 2 Layer > CNN 3 Layer > Evaluation
Evaluation Metrics	Accuracy, Precision, Recall, F-score
Results	Average score of CNN 3 layer-RNN Accuracy: 0.938 Precision: 0.997 Recall: 0.926 F-score: 0.960

In [6], a way of increasing accuracy and performance has been proposed. The paper identified relevant features inside the dataset, and the accuracy rate was improved. The researchers eliminated Redundant and irrelevant features which significantly improved classifier performance.

Table 3: Result Summary of Paper [6]

Language	English
Algorithms	Method "SelectPercentile" in the sklearn.feature_selection module
Dataset	KDD99
Methodology	1. Data Cleaning and Pre-processing 2. Features scaling 3. Features Selection 4. Model 5. Prediction and Evaluation
Evaluation Metrics	10-fold cross-validation
Results	Performance evaluation with 41 features for Dos Accuracy: 99.66 Precision: 99.505 Recall: 99.71 F-measure: 99.61 Performance evaluation with selected features for Dos Accuracy: 99.90 Precision: 99.69 Recall: 99.79 F-measure: 99.74

Finally, in [7], the authors reduced the amount of data in the NSL-KDD dataset by 80.4%, which reduced training time by 40% and testing time by 70%.

We have constructed our intrusion detection system using CNN and the NSL-KDD dataset. Then we compared the accuracy of other machine learning models when applied to this dataset.

4 DATA CLUSTERING

The current techniques of intrusion detection based on deep learning usually map the one-dimensional numerical features which are preprocessed into two-dimensional matrices. The redundant parts are filled with zeros. However, this method ignores a critical issue - the added relevance. The 2D matrix that has been transformed corresponds to a greyscale image and will impose a correlation around the matrix elements which will impact the training of the model and weaken its adaptability. In order to reduce this issue, we propose a method - data clustering.

If data contains 'm' features, we divide it into 'n' marts according to the previous knowledge in such a way that 'm' > 'n'. Then, we process the various parts of this data. Due to the correlation that exists between features, we divide the features data into 4 parts.

- Basic features
- Contents Features
- Time-based network statistics
- Host-based network traffic statistics

Data clustering helps to learn the relationships that exist between features which would otherwise be ignored if other classification algorithms are used. Since we are separating the features which are less relevant, the effects of correlations are significantly reduced.

5 CONVERSION TO MATRIX

The capability of CNN is better for processing images than other machine learning algorithms. So, we convert the inputs of our data into a form which represents images. We exploit the advantages of CNNs by transforming our network intrusion detection problem into an image classification problem. We convert the first part of the dataset, which contains 90 features, into a 9x10 matrix. In the rest of the data, the number of features is too small. So we cannot directly convert them into a matrix.

Therefore, we propose a method of transforming these data into a matrix. We do so using randomly repeated features. To be more precise, with 'n' features, we construct a 'n' x 'n' matrix using a random method. The method is such that at each position in the matrix, it makes each feature appear. By using this method, the network gains better structural information about the features because of effectively extracting the correlations between the features.

6 TRAINING MODEL

We use the same CNN structure for the various sections of the dataset. Taking the first part of the dataset as an example, the CNN model architecture that is implemented in the binary classification for intrusion detection and the dimensions of each layer are shown in Figure 1. The single model consists of an input layer, two layers of convolution and pooling, three layers that are fully linked and an output layer. Between the flattened model and the first fully connected layer to control over-fitting, a dropout layer with a 0.5 dropout rate is adopted for regularization.Each hidden layer has a Rectified Linear Unit (ReLU) activation function. The input layer transforms the data from the flow into a 2-dimensional matrix. The second layer is the convolution layer, and 32 zero-padding filters are used to extract input data. The third layer is the max-pooling

layer, and with a step size of 1. 1, the data is checked with 2.2 downsampling. The fourth layer is similar to the second, except that the number of kernels in the convolution is 64.

Figure 1: Block diagram of the proposed IDS.
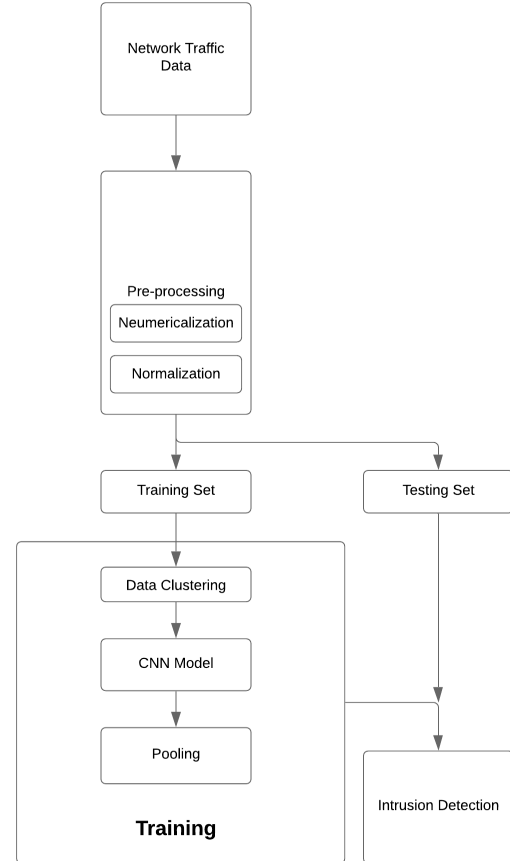
As the output of the classifier, a softmax layer in the last is used and notes that for the binary and multiclass classifications, the layer is distinct. The cost function of cross-entropy is used as the loss function to be minimized through training. 128, Adam and 0.001, respectively, are the batch size, the optimizer, and the learning rate used in training the networks.

7 PROPOSED METHODOLOGIES

7.1 Convolutional Neural Network (CNN)

CNN has two operations which are convolution and pooling. Convolution changes input data to output using a set of kernels or filters. The produced output showcases the features of the input data. That is why the output is known as the feature map. An activation function processes the convolution output further and down-sampling trims off irrelevant data using pooling. Pooling removes glitches in the data. That is how the learning improves for the following layers.[8][9] CNN adjusts the kernels/filters using rounds after rounds of learning so that the feature map can functionally represent the input data. We use ID convolution since the

network packet in the dataset is represented in an ID format.

$$(f * g)(i) = \sum_m^{j=1} g(j).f(i - j + m/2)$$

Here, i denotes the position of the values in sequence data. The activation function is ReLU. Finally, we use max pooling.

Figure 2: An example of the architecture of a single CNN model
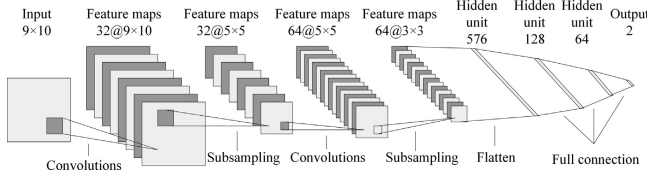
In a traditional neural network, the data first goes to the input layer. Then it goes to the hidden layer and from there to the output layer. The layers are fully connected, and in the same layer, there is no connection between nodes. Traditional neural networks, therefore, have many issues that can not be solved. The architecture of convolutional neural networks [12] is an improvement over the architecture of standard neural networks. CNN has achieved extraordinary results in fields such as image classification and speech analysis because of it [13].

In CNN, there are:

(1) One or more convolutional layers
(2) Pooling layers at the top
(3) Fully connected layers
(4) Dropout layers which serve as regularization layers. [14]

Because of this structure, CNN can take advantage of the 2D structure of the input data. As such a network can take an image as an input. Therefore, we avoid complicated feature extraction and unnecessary data reconstruction of traditional recognition algorithms. The modelling efficiency can be increased, and the difficulty of processing data manually can be decreased through pooling, shared weights and sparse connectivity. CNN can learn from various levels of features from a vast amount of data that is unlabeled. Therefore, the ways CNN can be used in a field such as a network intrusion detection are comprehensive. The architecture of a method similar to the proposed method used in this paper is shown in Fig. 2. [18]

8 DATASET

The dataset NSL-KDD is not the first of its kind. The KDD Cup was a competition for International Knowledge Discovery and Data Mining Tools. This competition was held in 1999 to gather traffic records. A network intrusion detector, a predictive model capable of distinguishing between "bad" connections, called intrusions or attacks, and "good" regular connections, was the competition task. A massive amount of internet traffic records were collected and bundled into a data set called the KDD'99 as a result of this competition. From this, the NSL-KDD data set was created as a revised, cleaned-up version of the University of New Brunswick KDD'99.

Four subsets of data are included in this data set: KDDTest+, KDDTest-21, KDDTrain+, KDDTrain+ 20Percent. However, KDDTest-21 and KDDTrain+ 20Percent are subsets of KDDTrain+ and KDDTest+. KDDTrain+ will be referred to as 'train' from now on, while KDDTest+ will be referred to as 'test'. Without the most difficult traffic records, the KDDTest-21 is a sub-set of test (Score of 21). The KDDTrain+ 20Percent is a sub-set of train whose record count makes up 20% of the entire dataset of train. That being said, the traffic records existing in the KDDTest-21 and KDDTrain+ 20Percent are already in test and train respectively, and there are no new records held from either dataset.

These data sets contain internet traffic records seen by a simple intrusion detection network and are the ghosts of the traffic that a real IDS encounters and there are only traces of its existence. There are 43 features per record in the dataset. 41 of the features are traffic inputs and the last two are labels. The first label indicates whether the data represents a normal packet or an attack packet. The last label represents the score (the severity of the traffic input itself).

There are four different attack classes within the data set: User to Root (U2R), Denial of Service (DoS), Probe, and Remote to Local (R2L). Here is a brief description of each attack below:

- DoS is an attack which attempts to shut down the flow of traffic from and to the target system. An abnormal amount of traffic is flooded with the IDS, which the system can not handle, and shuts down to protect itself. This prevents a network from being visited by normal traffic. An example of this could be that on a day with a big sale, an online retailer gets flooded with online orders, and because the network can not handle all the requests, it will shut down to avoid paying customers to buy anything. In the data set, this is the most prevalent attack.
- A probe is an attack that attempts to get data from a network. The objective here is to behave like a thief and steal critical data, whether it be personal customer information or banking information.
- U2R is an attack that starts with a regular user account and tries, as a super-user (root), to gain access to the system or network. To gain root privileges/access, the attacker attempts to exploit the vulnerabilities in a system.
- R2L is an attack that attempts to gain a remote machine's local access. An attacker does not have access to the system/network locally and attempts to "hack" their way into the network.

We can see that DoS is different from the other three attacks. DoS tries to shut down a system to stop traffic flow completely. In contrast, the other three attempts to infiltrate the system undetected. There are:

- 11 types of DoS attack
- 6 types of Probe attacks
- 7 types of U2R attacks
- 15 types of R2L attacks

A breakdown of the various subclasses of each attack existing in the dataset is shown in table number 4.

Table 4: Multiclass Attacks in NSL-KDD

Classes	DoS	Probe	U2R	R2L
	apache2	ipsweep	buffer_overflow	ftp_write
	back	mscan	loadmodule	guess_passwd
	land	nmap	perl	httptunnel
	neptune	portsweep	ps	imap
	mailbomb	saint	rootkit	multihop
	pod	satan	sqlattack	named
	processtable		xterm	phf
Sub-Classes	smurf			sendmail
	teardrop			snmpgetattack
	udpstorm			spy
	worm			snmpguess
				warezclient
				warezmaster
				xlock
				xsnoop
Total	11	6	7	15

The distribution is strongly skewed, although these attacks exist in the data set. In the table below, a breakdown of the record distribution can be seen. More than half of the records in each data set are regular traffic, and the U2R and R2L distribution is shallow. While this is low, this is an accurate representation of the distribution of modern-day internet traffic attacks, where DoS and U2R and R2L are hardly ever seen as the most common attack. The traffic record

Table 5: Number of records in NSL-KDD

Dataset	Number of Records					
	Total	Normal	DoS	Probe	U2R	R2L
KDD Train+	25192	13449	9234	2289	11	209
KDD Test+	125973	67343	45927	11656	52	995
KDD Test-21	22544	9711	7458	2421	200	2654

characteristics provide the IDS with information on the encounter with the traffic input. They can be split into four categories: Intrinsic, Content, Host-based, and Time-based. A description of the various feature categories is provided below:

- Without looking into the payload itself, intrinsic features can be derived from the packet's header and hold the necessary packet information. Features 1–9 are included in this category.
- Since the content features are sent in multiple pieces rather than one, they hold information about the original packets. The system can access the payload with this information. Features 10–22 are included in this category.
- Time-based features hold a two-second window to analyze the traffic input and contain information such as how many connections it was trying to make to the same host. Instead of data about the content of the traffic input, these features are mostly counts and rates. Features 23–31 are included in this category.

- Host-based features are comparable to Time-based features, except that it analyzes over a series of connections made (how many requests to the same host over x-number of connections) instead of analyzing over a 2-second window. These features are intended to access attacks that are longer than a window period of two seconds. Features 32–41 are included in this category.

It is possible to break down the feature types in this data set into four types.:

- Features: 2, 3, 4, 42 -> 4 Categorical
- Features: 7, 12, 14, 20, 21, 22 -> 6 Binary
- Features: 8, 9, 15, 23–41, 43 -> 23 Discrete
- Features: 1, 5, 6, 10, 11, 13, 16, 17, 18, 19 -> 10 Continuous

All the features are shown in the table below:

Table 6: Features of NSL-KDD

No.	Features	No.	Features
1	duration	22	is_guest_login
2	protocol_type	23	count
3	service	24	srv_count
4	flag	25	serror_rate
5	src_bytes	26	srv_serror_rate
6	dst_bytes	27	rerror_rate
7	land	28	srv_rerror_rate
8	wrong_fragment	29	same_srv_rate
9	urgent	30	diff_srv_rate
10	hot	31	srv_diff_host_rate
11	num_failed_logins	32	dst_host_count
12	logged_in	33	dst_host_srv_count
13	num_compromised	34	dst_host_same_srv_rate
14	root_shell	35	dst_host_diff_srv_rate
15	su_attempted	36	dst_host_same_src_port_rate
16	num_root	37	dst_host_srv_diff_host_rate
17	num_file_creations	38	dst_host_serror_rate
18	num_shells	39	dst_host_srv_serror_rate
19	num_access_files	40	dst_host_rerror_rate
20	num_outbound_cmds	41	dst_host_srv_rerror_rate
21	is_host_login		

The Flag is not very easy to understand, unlike the Protocol Type and Service whose values are self-explanatory (these values describe the connection). The Flag feature describes the connection's status and whether or not a flag has been raised. Each value in the Flag represents a connection's status.

9 DATA PREPROCESSING

9.1 Numericalization

In the NSL-KDD dataset, there are three features which are non-numeric and 38 numeric features. Since the input values must be numeric, we convert the non-numeric features into numeric. For example, the feature 'protocol_type' can have three different types of attributes which are 'tcp', 'udp', and 'icmp'. We encode them as binary vectors (0,0,1), (0,1,0) and (1,0,0). This way, we convert the 41-dimensional feature map into a 122-dimensional feature map.

9.2 Normalization

There are several features in the dataset in which the difference between the max and min values are large. Such features are dst_bytes[0,1.3 × 109], src_bytes[0,1.3 × 109] and duration[0,58329]. We apply the logarithmic scaling method to lower the differences and then use the formula below to map them to the [0,1] range:

$$x_i = (x_i - Min) / (Max - Min)$$

9.3 Feature Selection

The features in a traffic record provide the information about the encounter with the traffic input by the IDS and can be broken down into four categories: Intrinsic, Content, Host-based, and Time-based. We have considered the whole dataset for training as a four part dataset for classifying the attacks. There are many highly correlated features which hampers the model in classifying in the validation and testing time. So, we split the dataset in four part and took the first part as hybrid features.

Figure 3: Selecting the most relevant features of NSL-KDD
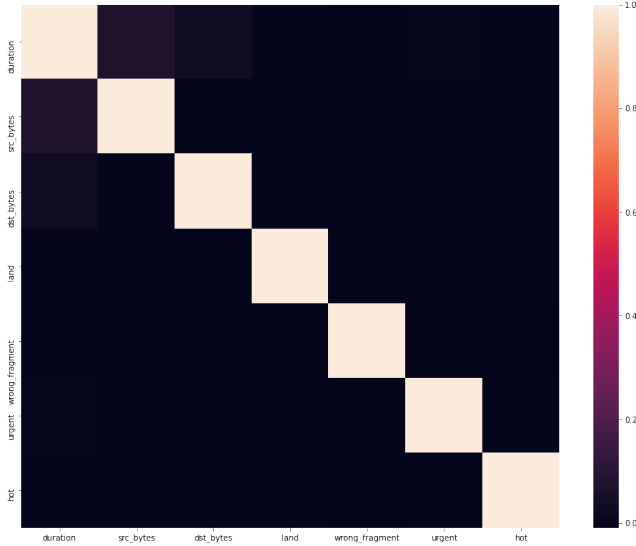
Intrinsic features can be derived from the header of the packet without looking into the payload itself, and hold the basic information about the packet. First 10 features of each packet gives the model more generalization and inference ability which helped us to go through all this way with a better accuracy than other models.

9.4 Evaluation Metrics

We use Accuracy (AC) to measure the performances of our model. Furthermore, we also introduce false positive rate and detection rate. True Positive (TP) denotes the number of records that rejects correctly and identifies as anomalies. Whereas, True Negative (TN) denotes the opposite. True Negative (TN) denotes the correct records that are normal and False Negative (FN) denotes the opposite.

We follow the notations given below:

Accuracy, AC = (TP + TN) / (TP + TN + FP + FN)
True Positive Rate, TPR = TP / (TP + FN)
False Positive Rate, FPR = FP / (FP + TN)
True Negative Rate, TNR = TN / (TN + FP)
False Negative Rate, FNR = FN / (FN + TP)
Sensitivity, TPR = TP / (TP + FN)
Specificity, FPR = TN / (TN + FP)

Therefore, our motivation is to get high accuracy and better detection rate with low false positive.

9.5 Binary Classification

We have mapped 41-dimensional features into 83-dimensional features. Therefore, the CNN-IDS model has 122 input nodes and 2 output nodes in the binary classification experiment. We take the number of epochs as 100 and the learning rate as 0.01. To find the better model, let the number of hidden nodes be 60, 80 and 120, respectively. The no. of hidden layers is 2, and the batch size is 64. From the table below, we have determined that a hidden node value of 64 achieves the best result.

Table 7: Confusion matrix of 2-category classification on KDDTest+

	Anomaly	Normal
Anomaly	8721	990
Normal	2576	10238

The experiments show that the CNN-IDS model works with a good detection rate (84.16%) when given 100 epoch for the KDDTrain dataset. The results obtained by J48, Naive Bayesian, Random Forest, Multi-layer Perceptron, Support Vector Machine and the other classification algorithms, and the artificial neural network algorithm also gives 81.2%, which is the recent literature about ANN algorithms applied in the filed of intrusion detection. Fortunately, these results are all based on the same benchmark - the NSL-KDD dataset. In [3], the maximum accuracy that the authors have achieved using the traditional methods is 99.40%. The

Table 8: Convolutional Neural Network Accuracy for Binary Classification

Our model		
Train	Validation	Test
96.22%	84.16%	70.26%

performance of CNN-IDS model is superior to other classification algorithms in binary classification.

9.6 Multiclass Classification

Inspired by [4], We have used both Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) for multiclass classification. In CNN, the number of hidden layers that we have taken is 2. The number of epochs is 15 and 150 respectively. In RNN, almost

all the values are similar to those used in the binary classification experiment. The difference is that the number of hidden nodes is taken as 80 and 120 respectively. It is observable from the table below that CNN with the number of epochs being 150 gains the best results.

Table 9: Confusion Matrix for Multiclass Classification

	Normal	DoS	R2L	U2R	Probe
Normal	9017	633	53	2	6
DoS	126	980	0	0	0
R2L	93	101	5528	0	0
U2R	33	0	0	2	2
Probe	1823	0	0	0	376

In this experiment,the detection rate of the CNN-IDS model gets higher accuracy testing dataset, not only higher than the detection rate on the NSL-KDD dataset, but also higher than other neural network models. The experimental results show that the fully connected model has stronger modeling ability and higher detection rate than the reduced-size CNN model.

Table 10: CNN Model Accuracy for Multiclass Classification

Our model		
Train	Validation	Test
99.11	84.70	65.45

The accuracy of 99.11 that this model has achieved is better than both [4]'s 93.8%.

10 EXPERIMENT RESULTS & DISCUSSION

We have used Google Collab's Free Tier plan to conduct all of our experiments. We use two experiments to study the performance of the IDS model. First, binary classification. And secondly, five-category classification such as Normal, DoS, R2L, U2R and Probe.

Table 11: Performance of the CNN model and other traditional machine learning models in binary classification.

	KDDTest+	KDDTest-21
Naive Bayes	76.56	55.77
J48	81.05	63.97
NB Tree	82.02	66.16
Random Forest	80.67	63.26
SVM	69.52	42.29
RNN	83.28	68.55
Our Model CNN	84.16	70.26

In order to compare the performance of different classification algorithms on the benchmark dataset for the multicalss classification as the binary classification experiments J48, Naive Bayesian, Random Forest, Multi-layer Perceptron, Support Vector Machine and other machine learning algorithms are used to train models through

the training set (using 10-layer cross-validation) by mean of the opensource machine learning and data mining software Weka.

Figure 4: Binary Classification
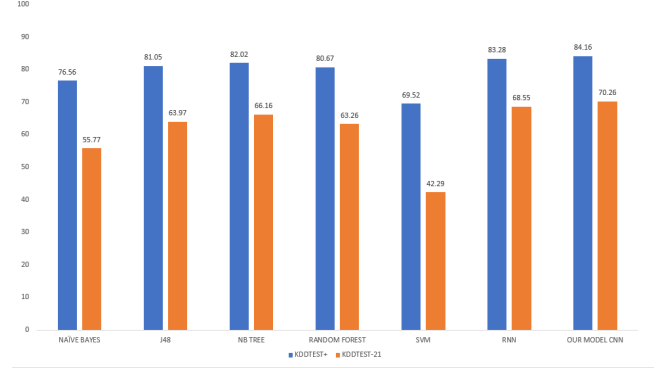
We then apply the models to the testing set. The results are described in Fig. 4. Compared with the binary classification, the accuracy of classification algorithms is declined in the five-category classification.

Table 9 shows the confusion matrix of the CNN-IDS on the test set KDDTest+ in the five-category classification experiments. The experiment shows that the accuracy of the model is 84.70% for the test set KDDTest+ and 65.45% for KDDTest-21, which is better than those obtained using J48, Naive bayes, Random Forest, Multi-layer Perceptron and the other classification algorithms.

In the binary classification experiments, we have compared the performance with Naive Bayes, Support Vector machine, Random Forest and K-Nearest Neighbors models.

Table 12: Performance of the CNN model and other traditional machine learning models in multiclass classification.

	KDDTest+	KDDTest-21
Naive Bayes	74.40%	55.77%
J48	74.60%	51.90%
NB Tree	75.40%	55.77%
Random Forest	74.00%	50.80%
SVM	74.00%	50.70%
RNN	81.29%	64.67%
Our Model CNN	84.70%	65.45%

We have evaluated the performance of different classification and neural network models to compare with our proposed model and we have found that we have achieved better result then other model by using Adam optimizer and epoch number 500. We have compared our model with other existing models and found that our model achieves better than any other neural network model in intrusion detection system but classification algorithms perform better for this specific problem because of labelled data and for the structure of the dataset.

Figure 5: Multiclass Classification
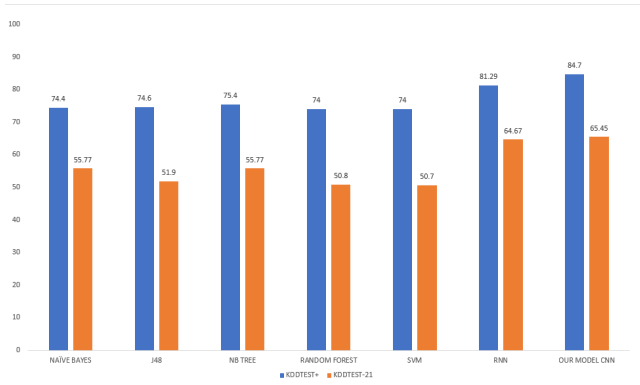
In the same way, we analyze the performance of multi-classification of the CNN-IDS model against Naive Bayes, Support vector machine, Random Forest and K-Nearest Neighbors and RNN. Based on the same benchmark, using KDDTrain+ as the training set and KDDTest+ and KDDTest-21 as testing set, the experimental result shows that both binary classification and multiclass classification the intrusion detection model of CNN-IDS has better test result and accuracy than the other machine learning models and maintains a high accuracy rate, even in the multiclass classification problem. Of course, the model we proposed will spend more time for training, but using GPU acceleration can reduce the training time.

11 CONCLUSION

Our experiment results say that in both binary classification and multiclass classification, the intrusion detection model using neural networks achieve higher accuracy then tradition machine learning models using the same dataset. Though our models require more computation time, additional hardware can reduce that to a great extent.

12 FURTHER RESEARCH

Using the research done in paper [6], redundant and irrelevant features can be removed, which can significantly improve classifier performance. By identifying relevant features inside the dataset, accuracy increases. Furthermore, the authors of [4] suggested the use of UNSW-NB15, which removes the inherited issues of KDDCup 99 and NSL-KDD.[7] Finally, Principal component analysis (PCA) can be used, as shown in paper [7], to drastically reduce the number of features, training and testing time. Combining these three ideas, we believe that we can get further accuracy and performance improvements to our models.

REFERENCES

[1] I Sumaiya Thaseen, B Poorva, and P Sai Ushasree. Network intrusion detection using machine learning techniques. In 2020 International Conference on Emerging Trends in Information Technology and Engineering (ic-ETITE), pages 1–7. IEEE, 2020.

[2] A. Javaid, Q. Niyaz, W. Sun, and M. Alam, "A deep learning approach for network intrusion detection system," presented at the 9th EAI Int. Conf. Bio-inspired Inf. Commun. Technol. (BIONETICS), New York, NY, USA, May 2016, pp. 21–26.

[3] Kirty, A. D. (2018, March). Network Intrusion Detection using Machine Learning. Website. https://github.com/Anshumank399/Network-Intrusion-Detection-using-Machine-Learning/blob/master/Final%20Report.pdf

[4] R. Vinayakumar, K. P. Soman and P. Poornachandran, "Applying convolutional neural network for network intrusion detection", Proc. Int. Conf. Adv. Comput. Commun. Inform., pp. 1222-1228, 2017.

[5] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in 2015 Military Communications and Information Systems Conference (MilCIS), pp. 1–6, IEEE, 2015.

[6] Herve. Nkiama, Syed. Zainudeen Mohd Said and Muhammad. Saidu, "A Subset Feature Elimination Mechanism for Intrusion Detection System", International Journal of Advanced Computer Science and Applications, vol. 7, no. 4, 2016.

[7] S. Lakhina, S. Joseph and B. Verma, "Feature reduction using principal component analysis for effective anomaly–based intrusion detection on NSL-KDD", Int. J. Eng. Sci. Technol., vol. 2, no. 6, pp. 3175-3180, 2010.

[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, p. 436, 2015.

[9] Y. Bengio, I. Goodfellow, and A. Courville, Deep learning, vol. 1. Citeseer, 2017.

[10] NSL-KDD dataset. Website. https://www.unb.ca/cic/datasets/nsl.html

[11] Moustafa, N., & Slay, J. (2015b). UNSW-NB15: A comprehensive data set for network intrusion detection. 2015 Military Communications and Information Systems Conference. Canberra, Australia: MilCIS 2015-IEEE Stream.

[12] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, et al., Gradient-based learning applied to document recognition, Proc. IEEE 86 (11) (1998) 2278–2324.

[13] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, et al., Recent advances in convolutional neural networks, Pattern Recogn. 77 (2018) 354–377.

[14] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, J. Mach. Learn. Res. 15 (1) (2014) 1929–1958.

[15] Wenjie Zhang, Dezhi Han, Kuan-Ching Li, Francisco Isidro Massetto. "Wireless sensor network intrusion detection system based on MK-ELM" , Soft Computing, 2020

[16] "Cloud Computing – CLOUD 2018" , Springer Science and Business Media LLC, 2018

[17] Saporito, Gerry. "A Deeper Dive into the NSL-KDD Data Set" Medium, https://towardsdatascience.com/a-deeper-dive-into-the-nsl-kdd-data-set-15c753364657. Accessed 17 September 2019.

[18] Y. Li, Y. Xu, Z. Liu, H. Hou, Y. Zheng, Y. Xin, et al., "Robust detection for network intrusion of industrial IoT based on multi-CNN fusion", Measurement, vol. 154, Mar. 2020.

[19] C. L. Yin, Y. F. Zhu, J. L. Fei and X. Z. He, "A deep learning approach for intrusion detection using recurrent neural networks", IEEE Access, vol. 5, pp. 21954-21961, 2017.

[20] Filiz Türkoğlu. "Author Attribution of Turkish Texts by Feature Mining" , Lecture Notes in Computer Science, 2007

[21] Md. Abdul Wahab. "Strain effect on single and double walled carbon nanotubes" , TENCON 2009 - 2009 IEEE Region 10 Conference, 2009

[22] Chidchanok Lursinsap. "Decision tree induction based on minority entropy for the class imbalance problem" , Pattern Analysis and Applications, 2016

[23] Yang Xin, Lingshuang Kong, Zhi Liu, Yuling Chen, Yanmiao Li, Hongliang Zhu, Mingcheng Gao, Haixia Hou, Chunhua Wang. "Machine Learning and Deep Learning Methods for Cybersecurity" , IEEE Access, 2018

[24] Herve Nkiama, Syed Zainudeen, Muhammad Saidu. "A Subset Feature Elimination Mechanism for Intrusion Detection System" , International Journal of Advanced Computer Science and Applications, 2016