https://dx.doi.org/10.4314/dujopas.v7i3a.12
ISSN (Print): 2476-8316
ISSN (Online): 2635-3490
Dutse Journal of Pure and Applied Sciences (DUJOPAS), *Vol. 7 No. 3a September 2021*

A Predictive Model for Network Intrusion Detection System Using Deep Neural Network

[1]Aminuddeen Abubakar, [2]Ahmed Baita Garko

Department of Computer Science,
Faculty of Computing,
Federal University Dutse.

Email: [1]aminuddeen@fud.edu.ng, [2]abgarko@fud.edu.ng

Abstract
Network Intrusion Detection System (NIDS) is an important part of Cyber safety and security. It plays a key role in all networked ICT systems in detecting rampant attacks such as Denial of Service (DoS) and ransom ware attacks. Existing methods are inadequate in terms of accuracy detection of attacks. However, the requirement for high accuracy detection of attacks using Deep Neural Network requires expensive computing resources which in turn make most organisations, and individuals shy away from it. This study therefore aims at designing a predictive model for network intrusion detection using deep neural networks with very limited computing resources. The study adopted Cross Industry Standard Process for Data Mining (CRISP-DM) as one of the formal methodologies and python was used for both testing and training, using crucial parameters such as the learning rate, number of epochs, neurons and hidden layers which greatly determined the accuracy level of the DNN algorithm. These parameters were experimented with values that are lesser compared to previous studies, training and evaluation were also done on the KDD99 data-set. The varying values of accuracy obtained from this study on four models with different numbers of layers of 50-epochs and learning rate of 0.01 achieved competitive results in comparison with the previous research of 100-1000 epochs and learning rate of 0.1. Therefore, the model with two layers attained same accuracy of 0.955 as the model with three layers from the previous study out of the four models tested in this study.
Also, the models with three and four layers in this study attained an accuracy of 0.956, which is 0.001 greater than the previous study's models.

Keywords: Network-Based IDS, Host-Based IDS, Deep Neural Network, Denial of Service, Knowledge Discovery Dataset.

INTRODUCTION
The first step in Cyber Security is to identify threats and define a corresponding attacker model. Threats, including malware, physical damage or social engineering, can target the hardware, the network, the operating system, the application or the users themselves. Then, detection and prevention mechanisms must be defined to defend against these attacks. An earlier study shows that, a fair level of security can be provided by static defense mechanisms such as firewalls and software updates, more dynamic mechanisms such as intrusion detection systems (IDSs) should also be used (Gunes et al., 2005).

**Author for Correspondence*

According to Mofti-Rafie (2019) anti-threat applications such as antivirus software, firewalls, and spyware-detection programs became inadequate to detect and prevent the latest malicious activities resulting from advanced tools and techniques that are used by both attackers and intruders.

As opined by the above researchers, intrusion detections systems as dynamic mechanisms are the essentials in providing a reasonable level of security, by ensuring more effective solutions for detecting attacks as opine by Gunes et al., (2005). One of the core mechanisms to overcome these types of attacks is by using Intrusion Detection System (IDS) techniques.

Different detection techniques can be used in the data being monitored to search for the attack patterns. Signature-based detection systems attempt to find signatures of an attack in the resource being monitored, whereas, in the statistical anomaly based systems which is called Anomaly- based IDS. Pattern of the normal behaviors have stored in the IDS database. Any deviation from normal patterns is considered as an attack where the IDS generate an alarm to inform the Network Security Manager about the new detected attacks for immediate action.

IDS are the new technologies that monitor the activities on network or on the specific devices like servers to detect and prevent unauthorized traffics. For example Snort IDS software, theses Intrusion Detection Systems are very much like a CCTV camera above a business entrance or sensors on its doors. It is a passive system that scans incoming data traffic, once the IDS identified any dangerous or suspicious traffic, it can send alert or alarm to the NSM for further action.

Intrusion Detection Systems are typically classified as host-based, network-based and wireless IDSs. A host-based IDS monitor data such as device logs, file systems, and disk resources, whereas a network-based IDS only monitors network-based data. Wireless IDS is similar to NIDS in that it can examine wireless network traffics and analyze them to identify which external users are attempting to connect to an access point (AP) in order to engage in malicious activity (Gunes et al., 2005).

Neural Networks are a uniquely powerful tool in multiple class classification, especially when used in applications where formal analysis would be very difficult or even impossible, such as pattern recognition, nonlinear system identification, and control. Provided the neural network has been given sufficient time to train, the property of generalization ensures that the network will be able to classify patterns that have never been seen before. Similarly, the accuracy of classification problems depends on a variety of parameters, ranging from the architecture of the actual neural network to the training algorithm of choice. In this study, deep neural networks will be used to design a predictive model that will characterize both normal and attack behaviours from the training data as an adaptive network intrusion detection that predict attacks as efficient and effective as possible for network administrators.

Existing methods are inadequate in terms accuracy of detection of attacks for IDSs. However, the requirements for high accuracy detection of attacks using deep neural network requires expensive computing resources which in turn makes most organizations, and individuals shy away from it**.**

However, the requirement of expensive computing power that neural networks demand in order to achieve high performance result are what makes most of the market world, organizations and individuals with limited computing power shy away from it, this study therefore aims at designing a predictive model for network intrusion detection using deep neural networks with very limited computing resources, with the following objectives; Study the existing models for IDS with the intention of providing a better model, Design a DNN model that achieves high accuracy with lesser computing power/resources and Compare results obtained from this study with previous studies.

RELATED WORK

The Intrusion Detection System (IDS) is intended to be an additional measure of protection used to detect the attack by sending an alert or alarm for preventive action. likewise, an IDS is essentially a software application that tracks network or system activities and figures out if any malicious operations occur, using various types of strategies, techniques, methods, and algorithms. If the intrusion is detected more quickly, allowing the network administrator to identify the type of attack early and select appropriate protective mechanisms to prevent the threat from affecting the systems, resources, and causing any damage to the systems, a suitable selection of IDS location types, techniques and deployment methodology should be considered.

TYPES OF INTRUSION DETECTION SYSTEMS (IDS)

A. **Network-Based IDS**
According to Modi & Jain (2016), NIDS is a system that observes network traffic usually consisting of sensors distributed over the network and a processing unit. The sensors sniff network packets, for example TCP/IP packets, and the scheme efforts to recognize malicious packets or abnormal action on the network. In this type of IDS, the device sensor was placed behind the external firewall or is more precisely identified at a boundary between networks such as in routers, firewalls, and virtual private networks to detect threat from outside worlds and attempt to penetrate the perimeter defense of the network.

B. **Host-Based IDS**
Host-based IDS (HIDS) is a technology or scheme that exists on the local computer as an agent or host and observes the behavior of the machine, for example by examining logs (Modi & Jain 2016). Similarly, the software is installed on each of the computer hosts of the network to monitor the events occurring within that host only not the whole network.

C. **Wireless IDS**
A wireless local area network intrusion detection system is similar to NIDS in which it can examine wireless network traffics and analyze them to identify which external users that trying to connect to access point (AP) to cause any types of malicious activities. This type of IDS is developed into access points or in wireless routers or behind the firewall.

IDS TECHNIQUES

Anomaly-Based Detection

Anomaly based detection is based on defining the activities in the network. Network activities are the predefined when accepted or activates the event in the detection of anomaly (Modi & Jain 2016). The network's recognized activities are arranged or learnt by the network manager's specifications.

An IDS that looks at network traffic and detects wrong or generally irregular data in this form of detection is called anomaly-based detection. This system helps to identify

unnecessary traffic which is not clearly identified. For example, anomaly-based IDS will detect mal-forming of an Internet Protocol (IP) packet. It does not detect that it is specifically malformed but rather indicates that it is anomaly. Similarly, it compares descriptions of what behavior, is considered normal to detect major deviations towards observed events. This method uses profiles that are developed by monitoring typical activity characteristics over a certain period of time (Mofti-Rafie, 2019).
As stated by Mofti-Rafie, (2019), one of the benefit of using an anomaly-based detection method is that it can be very successful in detecting previously unknown threats, but the common anomaly-based detection problems are when malicious activity is included within a profile, profiles are not complex enough to reflect real-world computing activity, and many false positive alarms are generated.

Signature-Based Detection
A signature is a prototype that corresponds to a recognized attacked or threat, and misuse detection is a method of comparing prototypes to captured occurrences to identify likely intrusions. It is making similar efforts as most antivirus programs do (Modi & Jain 2016). It checks the network for behavior that is preset to be malicious. They are flawless and swift as they only make a comparison between what they experiences and a predetermined law. Signature based IDS will not identify the most recent threats.

Deep Neural Network
Traditionally, machine learning algorithms are linear, and deep neural networks are stacked in increasing hierarchy of complexity as well as abstraction. Deep Neural Networks (DNNs) are nothing but Artificial Neural Network (ANN) with multi-layered structures constituted within the input and output layers. Similarly, they can model convoluted non-linear relationships and can render computational models where the object is expressed in terms of the layered arrangement of primitives (Rahul-Vigneswaran et al., 2018). DNN is a neural network with a certain level of complexity, a neural network with more than two layers. Deep neural networks use sophisticated mathematical modeling to process data in complex ways. Each layer applies a nonlinear transformation onto its input and creates a statistical model as output from what it learns. Deep Neural Networks (DNNs) as networks that have an input layer, an output layer and at least one or more hidden layers in between as it is shown in figure 2.1.

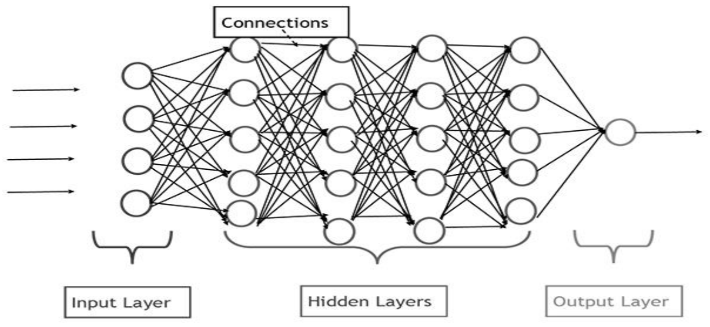


Figure 2.1: Typical DNN with four layers in the hidden layer (Szegedy C., *et al*, 2013)

The benefit of NNs according to Ahmad et al., (2015), Flexibility is the most significant advantage in the identification of a neural network. That the Neural Network is able to analyze network data, even if the data is incomplete or skewed. The Neural Networks have other advantages: speed. Since the output of a Neural Network is expressed as a probability, the Neural Network provides a predictive ability to detect any attacks. Similarly, one of the

another significant advantage of Neural Networks is the Neural Network's ability to "know" the characteristics of any attacks and identify them according to types of attack.
In terms of limitations, the Neural Networks have several drawbacks that contribute to the unusual detection usage, one of which is the Neural Network training requirements. Because of the complexity of the training approach utilized, which necessitates a significant amount of data and several revisions in order to provide reliable results. To ensure the results are statistically accurate, the training routine requires a very large amount of data. Nevertheless, the "black box" is the most important and recent downside of applying Neural Networks to intrusion detection. The "Black Box Problem" is now an ongoing area of research on the Neural Network (Asma et al., 2015);

Table 2.1: Summary of Related Work

S/No	Authors	Title	Techniques	Strengths	Weaknesses
1	Rahul-Vigneswaran et al. (2018).	Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security	Deep Neural Networks and Classical Machine Learning Algorithms using high-end GPUs and enough computing resources. i.e; fast CPU, SSD storage, fast and large Random Access Memory are all also required.	DNN has outperformed all the other classical machine learning algorithms at layer 3 within the performance metrics.	Training requirements and flexibility of the DNNs in adversarial environments are required so also the Black box problem.
2	Pradhny and Roychaudhary, 2016.	Discrimination Prevention in Data Mining for Intrusion and Crime Detection	Back propagation neural networks and Extreme Learning Machine reasonable computing power.	Proposed hybrid approach for intrusion detection that based on using two a new learning methodology towards developing a novel IDS system.	The proposed system is not easy to configure and it require more training time but it has self-adaptive learning.
3	Ahmad A. and Bhanu S., 2017.	Intrusion Detection System Based on Support Vector Machine Using BAT Algorithm	Support Vector Machine using BAT Algorithm using decent computing resources.	Proposed intrusion detection system that improves the detection accuracy and reduces false alarm rate with accuracy reaches up to 94.3%	It requires enhanced the accuracy to 94.3% for detecting known attack but it doesn't detect unknown attacks, It also requires good preprocessed dataset to work well.
4	Asma et al. 2015	Intrusion Detection Using Neural Network: A Literature Review	Artificial Neural Network using high computing power.	Flexibility is the most important advantage of a neural network in the detection. That the Neural Network can analyze the data from the network, even if the data is incomplete or distorted. The Neural Networks has other advantages that are the speed	Training requirement because of the complexity of the training method that were used, and need a very large amount of data and multi refinements to get accurate results.

It has been observed from reviewed literatures that achieving high accuracy detection IDS, the use of high-end GPUs, fast CPU, SSD storage, and fast and large Random Access Memory are all also required. Therefore, this study therefore aims at designing a predictive model for network intrusion detection using deep neural networks with very limited computing resources.

METHODOLOGY

The approach and tools used to construct the suggested model for an instruction detection system are discussed in this section. The goal of the research is to create a predictive model for network intrusion detection using deep neural networks with very limited computing resources, based on a set of parameters that are fed into the Deep Neural Networks algorithm on the data set, with the goal of achieving significant accuracy, performance, and detection rates. The concept of this methodology is shown in following figure 3.1 below.

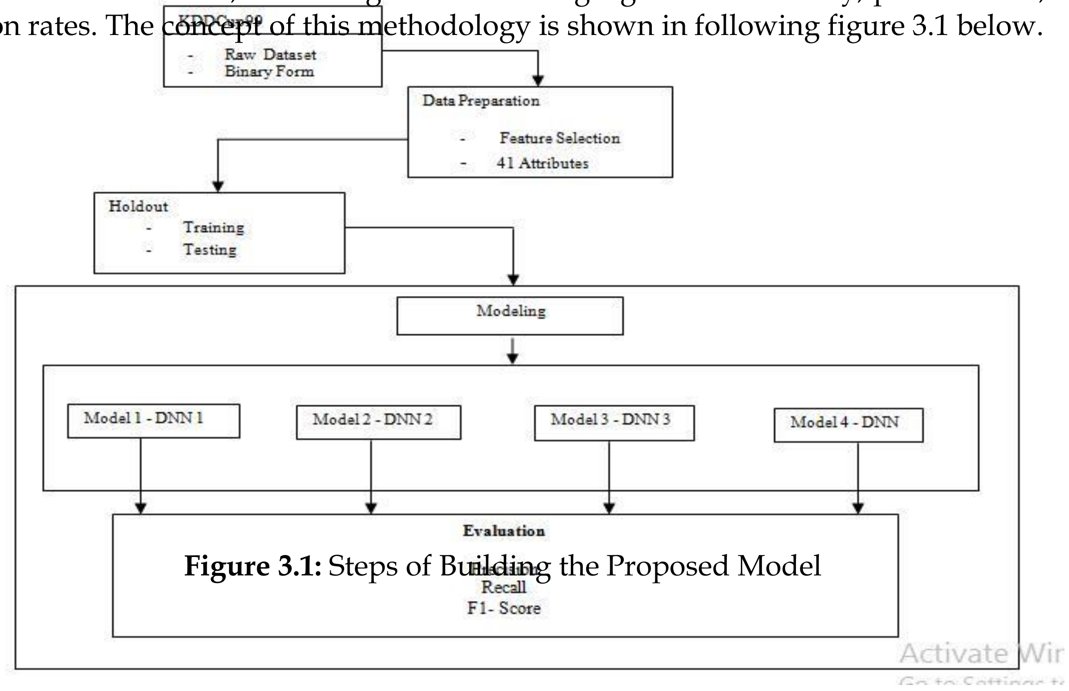


Figure 3.1: Steps of Building the Proposed Model

Several formal methodologies are available for implementing data mining tasks such as detection, classification, and prediction. In a study by Bhagwan (2018), the Cross Industry Standard Process for Data Mining (CRISP-DM) is one of the formal methodologies and it is adopted for modeling this study. It is a methodology that is widely used in classification and prediction activities as it's one of the most reliable and user-friendly technique.

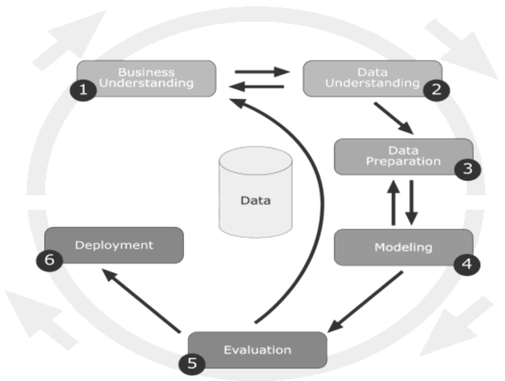


Figure 3.2: Methodology Steps of the CRISP-DM (Bhagwan., 2018)

CRISP-DM as shown in figure 3. 2 is based on six stages i.e. business and data understanding, then data preparation and modeling, and then on to evaluation specifically for this study.

BUSINESS UNDERSTANDING

The main focus of this study as regards to business understanding is to design a detective model for network intrusion. The model designed as Intrusion Detection System is aimed at classifying network traffic into:

i. Normal: The output is normal when no attack of any form is detected from the traffic data.
ii. Attack: The output is an attack when any of the following types of attacks is detected from the network traffic.
   a. Denial of Service (DoS) such as e.g. syn flood
   b. User to Root (U2R): Unauthorized access to local super-user (root) privileges, e.g., various buffer-overflow attacks
   c. Root to Local (R2L): Unauthorized access from a remote machine, e.g. guessing password
   d. Probing: Surveillance and other probing, e.g., port scanning

DATA UNDERSTANDING

The data required for this study is network traffic data. Research shows that traffic data such as DARPA and KDD datasets are used in training NIDS models because they contain attributes with details vital to understanding network traffic (Rahul-Vigneswaran et al., 2018). The selected dataset for training the model in this study is the KDD99 dataset.

DATA PREPARATION

Data preparation tasks generally include pre-processing activities in order to clean the data and get it in the right format for modelling. Dataset pre-processing activities generally include normalization, discretization, and dimensionality reduction. However, no pre-processing activities were carried out on the dataset used in this study because the data was obtained in both raw and binary format but only the binary one was used as required for the modelling activities.

MODELING APPROACH

The identification and selection of modelling technique is one of the crucial first steps to take in the modelling process. This step is followed by the generation of test scenario for validating the model's quality. This study uses unsupervised learning method; the architecture chosen for this study is a MLP and back-propagation mechanism which is used to train the DNN with fully connected hidden layers, which is appropriate starting point for many problems and especially predictions problems. Graphical representation of process model will be used to illustrate the working structure of the model using back-propagation for neural network generation.

As shown and discussed in the next chapter i.e. implementation, this study is on four different DNN models. The models are similar in terms of epoch number, learning rate or dropout and activation functions, while the difference between the different models is on the number of hidden layers. The differences between the different layers are elaborated in the experimental settings of the next chapter i.e implementation.

Modeling Tools

The modelling tools for this study are:

i. **KERAS**: This is an open source neural network library written in python. It is capable of running on top of TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or PlaidML. It is designed to enable fast experimentation with deep neural networks. It focuses on being user friendly, modular, and extensible. (Chollet, 2019). The modelling activities that would be carried out by Keras are: Data preparation, Activation functions and dropout rate, Data sequence reading, Deep Neural Network implementation, and Data writing in CSV.

Important features available in Keras that would facilitate this study:

- **a.** Keras contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, optimizers and a host of tools to make working with image and text data easier to simplify the coding necessary for writing DNN code.
- **b.** It support other common utility layers like dropout, batch normalisation, and pooling.

ii. **Scikit-Learn:** is a free software machine learning library for python programming language. It features various classification, regression and clustering algorithms. It is designed to interoperate with the python numerical and scientific libraries Numpy and Pandas. (Virtanen et al., 2019).

Important features available in Scikit Learn that would facilitate this study:

- a. It is built on top of several common data and math python that make it super easier to integrate between Numpy and Pandas and can pass numpy arrays and Pandas data frames directly to machine learning algorithms of scikit learn.
- b. It provides end-to-end training for both classification and regression. Similarly, layers can be easily defined in a tuple.

EVALUATION

Models evaluation would be based on standard data mining evaluation metrics. The metrics are as follows:

1. **Precision** - Precision is the ratio of correctly predicted positive observations to the total predicted positive observations. High precision relates to the low false positive rate. We have got 0.999 at layer one precision which is pretty good.
   Precision = TP/TP+FP
   Where,
   TP – denotes the number of connections classified as Normal while they actually were Normal.
   FP – denotes the number of connections classified as Attack while they actually were Normal.
2. **Recall: Recall** (Sensitivity) - Recall is the ratio of correctly predicted positive observations to the all observations in actual class - yes. The question recall answers are: We have got recall of 0.917 at layer two and four which is good for this model as it's above 0.5.
   Recall = TP/TP+FN
   Where,
   TP – denotes the number of connections classified as Normal while they actually were Normal.
   FN – denotes the number of connections classified as Normal while they actually were Attack
3. **F1-Score** - F1 Score is the weighted average of Precision and Recall. Therefore, this score takes both false positives and false negatives into account. Intuitively it is not as easy to understand as accuracy, but F1 is usually more useful than accuracy, especially if you have an uneven class distribution. Accuracy works best if false positives and false negatives have similar cost.

If the cost of false positives and false negatives are very different, it's better to look at both Precision and Recall. In our case, F1 score is 0.956.
F1 Score = 2*(Recall * Precision) / (Recall + Precision)

EXPERIMENTS SETTINGS

The experimental setting that guides the implementation of this study is shown in Table 4.1 below.

Table 4.1: Experiment Settings

Parameters	**DNN1**	**DNN2**	**DNN3**	**DNN4**
Input Layers	41	41	41	41
Hidden Layers	4	3	2	1
Output Layers	2	2	2	2
Activation Function	Non-linear i. ReLU – hidden layers ii. Sigmoid – Output layers			
Epochs	50	50	50	50
Regularization (Dropout rate)	0.01	0.01	0.01	0.01
Neurons	1024	1024 512	1024 512 256	1024 512 256 128

The details of the experimental settings are:

i. **Input Layers:** There are 41 input layers for all the Deep Neural Network experiment conducted. This number is based on the total number of attributes from the KDD99 network traffic dataset. All attributes from the dataset were used to serve as the input layers.

ii. **Hidden Layers:** The number of hidden layers is one of the features that separate the different experiments conducted. The varying numbers of hidden layers are one, two, three, and four. The different models based on the number of neurons per layer are as follows:

a) **Model 1:** DNN with one layer: The number of neurons for this model is 1024 neurons in the first layer which is the only layer.

b) **Model 2:** DNN with two layers: The number of neurons for this model is 1024 neurons for the first layer, 512 neurons for the second layer.

c) **Model 3:** DNN with three layers: The number of neurons for this model is 1024 neurons in the first layer, 512 neurons in the second layer, and 256 neurons in the third layer.

d) **Model 4:** DNN with four layers: The number of neurons for this model is 1024 neurons in the first layer, 512 neurons in the second layer, 256 neurons in the third layer, and 128 neurons in the fourth layers.

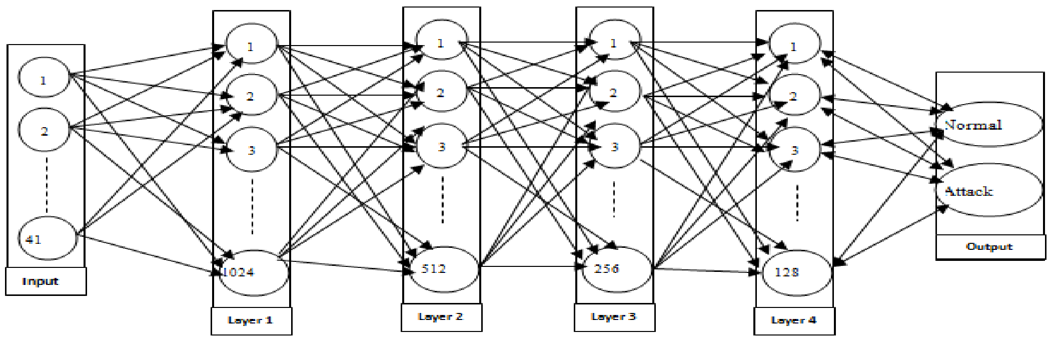


Figure 4.1: Architecture of the Proposed Model (Rahul-Vigneswaran et al., 2018)

iii. **Output Layers:** There are two (2) numbers of layers for all the DNNs model experiments conducted. The layers are 'normal' for all traffic data identified as normal, while 'attack' is for any network traffic identified as a threat.

iv. **Activation Functions:** In this particular case, this study adopts sigmoid and ReLU activation functions as a non-linear neuron activation function.

   a) **Rectified Linear Unit (ReLU):** In the input and hidden layers, this study adopted ReLU as non-linear activation function. Similarly, weights are added to the input signals (values) in order to support and fed them forward to the next hidden layer for abstract mathematical computation. The neuron count from first hidden layer to the last one was steadily increase in a bit-wise form in order to obtained more accurate result as opposed to the previous research.

   b) **Sigmoid:** In the output layer, sigmoid activation function squashes values between ranges of 0 to 1. The output consists of two only neurons as Normal and Attack. Since the neurons of the last hidden layers are multiples and must be converted to actualize the two actual outputs as Normal and Attack, sigmoid activation function was used as it's one of the natures to returns only two outputs.

v. **Number of Epochs:** It is the number of times that the model is exposed to the training dataset. In this study, the number of epoch used was 50 because of the less computing resources.

vi. **Learning Rate (Dropout):** Usually when training neural network, Gradient Descent is use to optimize the weights. Back-propagation was used to calculate the loss function's derivative with respect to each weight and subtract it at each iteration. Learning rate determines how quickly or how slowly you want to update your weight (parameter) value. Learning rate should be high enough so that it won't take ages to converge, and it should be low enough so that it finds the local minima. In this study a learning rate of 0.01 was used.

vii. **Neurons (Node):** Are the basic unit of a neural network. It gets certain number of inputs and a bias value. When a signal (value) arrives, it gets multiplied by a weight value. If a neuron has 4 inputs, it has 4 weight values which can be adjusted during training time.

The holdout approach is the implementation process used for the DNNs algorithm on the KDD dataset. The holdout approach is based on percentage split for training and testing. Table 4.2 shows the holdout implementation of this study.

Table 4.1: Holdout Percentage for Implementation

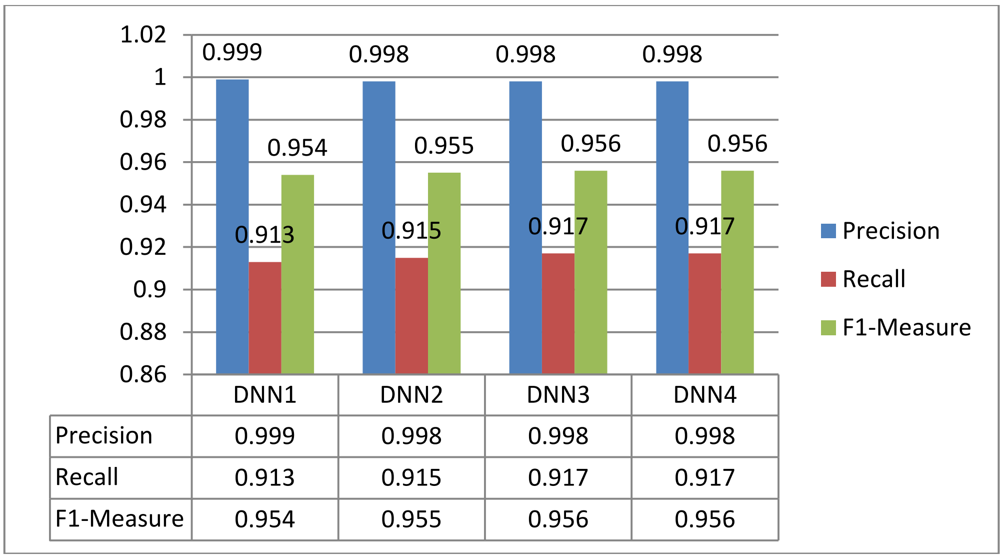
Training	Testing
70%	30%

RESULTS

The general accuracy of all four models were very close with very minimal margins across the different epochs experimented, therefore the comparative results analysis of the four models as shown in table 4.3 and figure 4.2 is discussed in this section with a view to provide clearer understanding into the varying accuracy in terms of: Precision, Recall, and F1-measure.

Table 4.2: Comparison of Results from four DNN models

Experiment	Precision	Recall	F1-Measure
DNN1	0.999	0.913	0.954
DNN2	0.998	0.915	0.955
DNN3	0.998	0.917	0.956
DNN4	0.998	0.917	0.956



	DNN1	DNN2	DNN3	DNN4
Precision	0.999	0.998	0.998	0.998
Recall	0.913	0.915	0.917	0.917
F1-Measure	0.954	0.955	0.956	0.956

Figure 4.2: A comparison of the Accuracy in-terms of Precision, Recall and F1-measure of the Four DNNs Layers.

As shown in figures 4.2 and table 4.3, all the four DNN models implemented in this study achieved a high result with a very slight difference across the different models. The last two models achieved the accuracy score in terms of precision, recall, and f1-measure i.e. 0.998, 0.917, and 0.956 respectively. This indicates that model 3 experiment parameters and model 4 experiment parameters as shown in table 4.2 and 4.3 respectively yielded same results of 50 epochs, 41 input layers, and 2 output layers. These same two models achieved the f1-measure of 0.956 which is the highest compared to other models followed by model 2 with 0.955, and model 1 with 0.954. Results also show that model 1 achieved the lowest in terms of f1-measure and recall i.e. 0.913 but scored the highest in terms of precision i.e. 0.999 compared to other models. Model 1, as the highest in terms of precision is followed by Models 2, 3, 4 which all achieved same precision of 0.998.

As varying accuracy scores are obtained for the different models shown in table 4.1, the accuracy values of F1-measure as the weighted mean precision and recall are used to mark the overall accuracy performance of the four models. Therefore, the model with the highest f1-measure which in this case are two models (i.e. models 3 and 4) with the same score of 0.956 are considered the best, followed by model 2 with 0.955, and model 1 with 0.954. However, it is important to highlight that the difference between all the models in terms of the f1-measure is not high as the difference between the highest and lowest is 0.002.

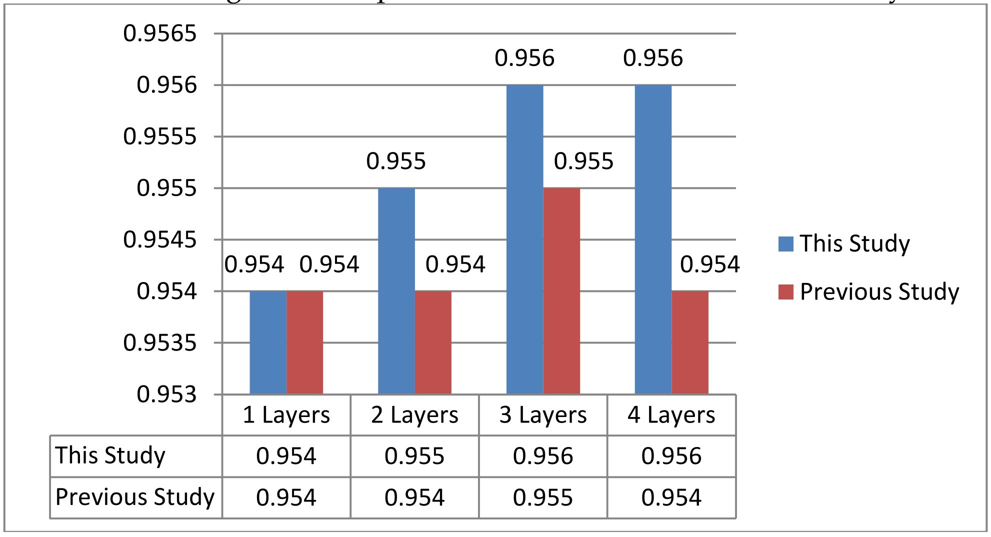
DISCUSSION

To further validate findings from this study, the results of previous study are compared with this study. This is comparison is done considering that this and the previous study are conducted on the same dataset i.e. KDD99 and same algorithms but with varying parameters as shown in table 4.3 below.

Table 4.3: Parameter Comparison with Previous Study

Parameter	**This Study**				**Previous Study** Rahul-Vigneswaran et al., **(2018)**				
Number of Epochs	50				100				
Dropout rate	0.01				0.1				
No. Hidden layers	1	2	3	4	1	2	3	4	5
No. Neurons per layer	1024	1024 512	1024 512 256	1024 512 256 128	1024	1024 768	1024 768 512	1024 768 512 256	1024 786 512 256 128

As the training is completed, results shows that same or even slightly improved result can be achieved with less number of epochs and learning rate. Table 4.3 illustrate the values of the result accuracy obtained at each training time of the DNN1, DNN2, DNN3, and DNN4 at 50 epochs and learning rate of 0.01 prior to the previous research of 100-1000 epochs and learning rate of 0.1 although less computational resources in the current study.



	1 Layers	2 Layers	3 Layers	4 Layers
This Study	0.954	0.955	0.956	0.956
Previous Study	0.954	0.954	0.955	0.954

Figure 4.3: Comparison of Results with Previous Study

From figure 4.3, a slightly significant improvement was achieved. Layer 3 and 4 has the highest accuracy in terms of f1-measure of 0.956 respectively. Contrarily, in terms of precision, layer 1 has the highest precision while layer 2, 3 and 4 has the same precision.

CONCLUSION

In this study, the researcher proved that despite limited computing resources, competitive accuracy values can be obtained. This is evident from the four models implemented with different numbers of layers but on 50-epochs and learning rate of 0.01. Out of the four models experimented in this study; the model with two layers achieved an accuracy of 0.955. Also, the models with three and four layers from this study achieved an accuracy of 0.956. Therefore, the achievement of this study regarding the accuracy of 0.956 and learning rate of 0.001 surpasses the result of the previous study with one, two, three, four, and five layers of the previous model and the layer with three layers was their best with 0.955 detection accuracy.

Although several achievements have been made in this research, there are recommendations for future work. They are:

i. Implementation of user interface program with text message and email alert notification on attack detection.
ii. As this research is scoped to network intrusion detection, the future work would be extended to host based intrusion detection.

REFERENCES

Ahmad A. and Bhanu-Pratap S. (2017).” Instruction Detection System Based on Support Vector Machine using BAT Algorithm” in international journal of computer applications, Vol. 158 – No 8, January 2017.

Asma, A. H. et al (2015). “Intrusion Detection Using Neural Network: A Literature Review” International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064. Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391.

Bhagwan, S. (2018). “CRISP Model: A Structured Approach for Presentation of Research.” CSI Communications, 6th International Conference on Innovations in Computer Science & Engineering (ICICSE-2018) ISSN 0970-647X, Volume No.42/Issue No. 7.

Chollet, F. (2018). Keras: “The python deep learning library”. *Astrophysics Source Code Library*, ascl-1806.

Gunes, H. Kayacik A., and Zincir-Heywood N.,Malcolm I. (2005). “Selecting Feature for Intrusion Detection: A Feature Relevance Analysis on KDD99 Intrusion Detection Datasets”. Third Annual Conference on Privacy, Security and Trust, 12-14, 2005, The Fairmount Algonquin, St. Andrews, New Brunswick, Canada, Proceedings.

Mofti-Rafie, A. A. (2019). “Enhancing Hybrid Intrusion Detection and Prevention System for Flooding Attacks Using Decision Tree”. A Thesis research, Sudan University of Science and Technology.

Modi, U. and Jain, A. (2016). “An Improved Method to Detect Intrusion”. *Inf. Eng.*

Pradhnya, K. and Roychaudhary, R. C (2016). “Discrimination Prevention in Data Mining for Intrusion and Crime Detection”, in International Research Journal of Engineering and Technology, Vol. 4, 2016.

Rahul-Vigneswaran, K. V. et al (2018). “Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security”. International Conference on Computing, Communication and Networking Technologies *(ICCCNT)*.

Szegedy, C. et al (2013). “Deep Neural Networks for Object Detection.” Advances in information Processing Systems. 2553-2561.

Virtanen, P., Gommers, R., Burovski, E., Oliphant, T.E., Cournapeau, D., Weckesser, W., Peterson, P., Mayorov, N., van der Walt, S., Wilson, J. and Laxalde, D., 2019. Scipy/scipy: SciPy (Version 1.2. 1. *Zenodo*) [Computer program].

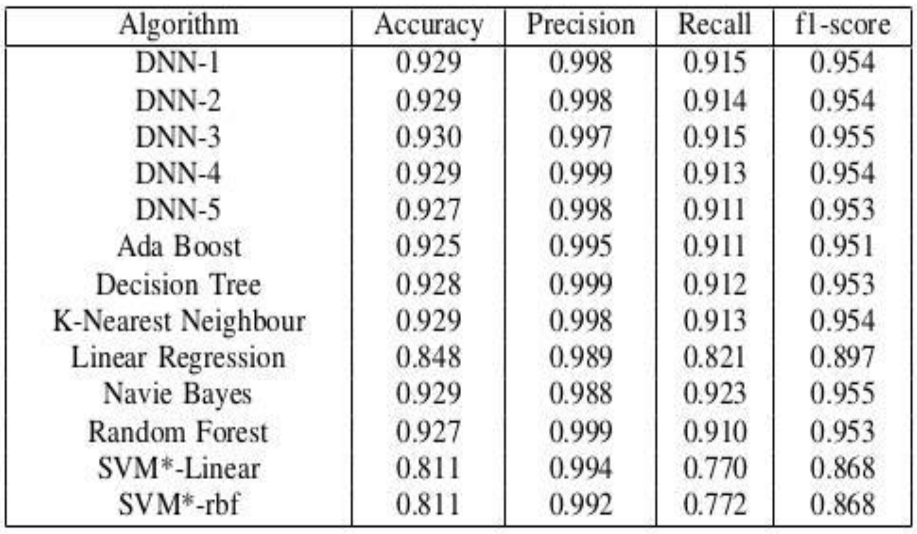
APPENDICES

Appendix A: Accuracy and Loss per Epoch on Training
Model Four – DNN4

Epoch	Accuracy	Loss
0	0.995753	0.013672
1	0.997622	0.00799
2	0.997913	0.006931
3	0.998259	0.005956
4	0.99848	0.005399
5	0.998593	0.004841
6	0.998751	0.004622
7	0.998842	0.004305
8	0.998786	0.004183
9	0.998866	0.00411
10	0.998964	0.003604
11	0.998941	0.003782
12	0.998919	0.003652
13	0.99899	0.003417
14	0.999055	0.003411
15	0.999069	0.003151
16	0.999071	0.003136
17	0.999095	0.002963
18	0.999128	0.003026
19	0.999136	0.002775
20	0.999162	0.002704
21	0.99914	0.002773
22	0.999188	0.002725
23	0.999164	0.002546
24	0.999235	0.002424
25	0.999233	0.003205
26	0.999227	0.002654
27	0.999294	0.002208
28	0.999269	0.002253
29	0.999279	0.002197
30	0.999247	0.002352
31	0.999336	0.002198
32	0.999324	0.00216
33	0.999267	0.002593
34	0.999377	0.002053
35	0.999391	0.002096
36	0.99935	0.002808
37	0.999383	0.001893

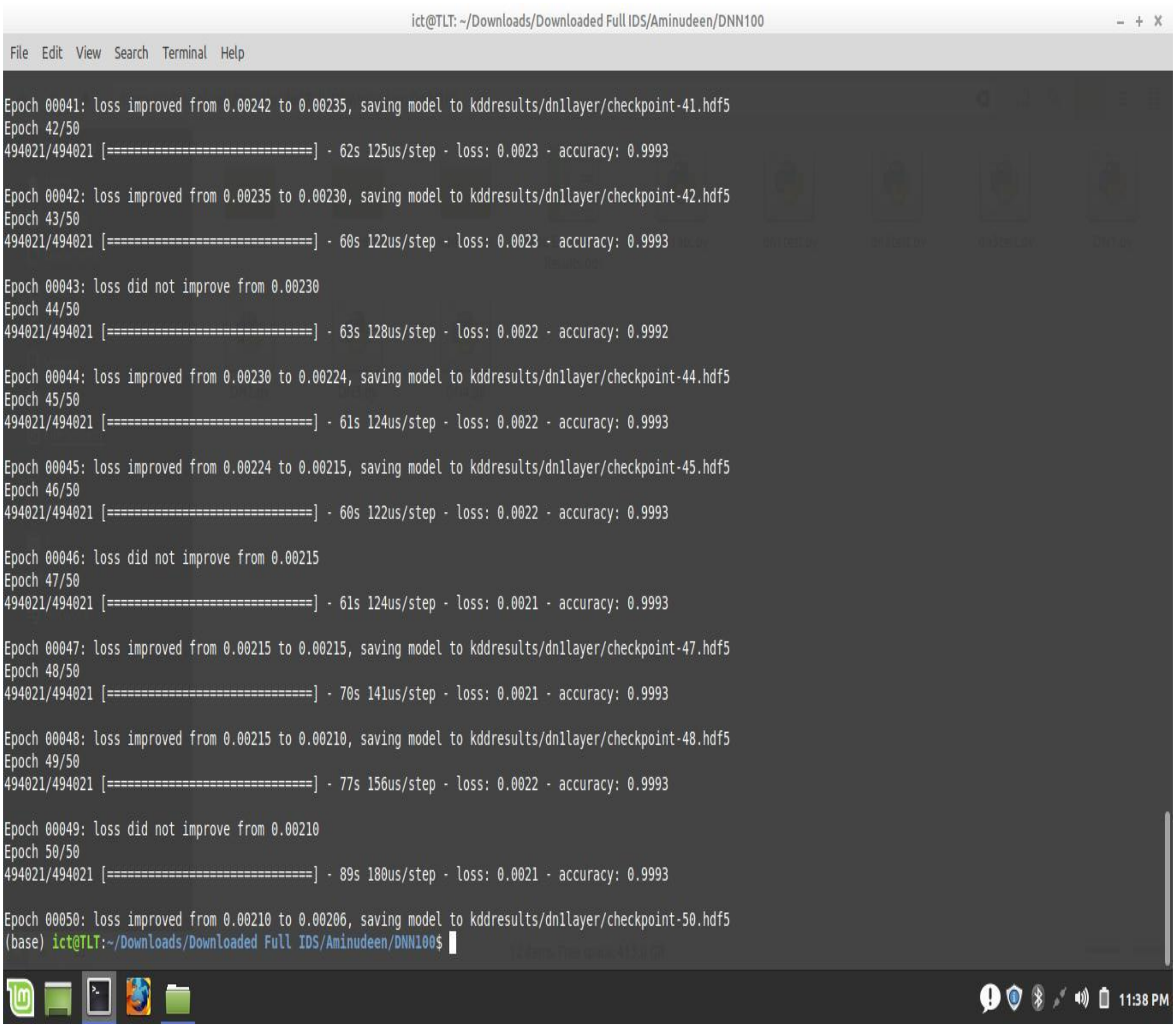
Appendix B: Result of Previous Study

Result of the Previous Study conducted by Rahul-Vigneswaran, K., Vinayakumar, R., Soman, K. P., & Prabaharan, P. (2018). On Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security.



Algorithm	Accuracy	Precision	Recall	f1-score
DNN-1	0.929	0.998	0.915	0.954
DNN-2	0.929	0.998	0.914	0.954
DNN-3	0.930	0.997	0.915	0.955
DNN-4	0.929	0.999	0.913	0.954
DNN-5	0.927	0.998	0.911	0.953
Ada Boost	0.925	0.995	0.911	0.951
Decision Tree	0.928	0.999	0.912	0.953
K-Nearest Neighbour	0.929	0.998	0.913	0.954
Linear Regression	0.848	0.989	0.821	0.897
Navie Bayes	0.929	0.988	0.923	0.955
Random Forest	0.927	0.999	0.910	0.953
SVM*-Linear	0.811	0.994	0.770	0.868
SVM*-rbf	0.811	0.992	0.772	0.868

Appendix C: Screen Shot of Accuracy and Loss per Epoch on Training


ict@TLT: ~/Downloads/Downloaded Full IDS/Aminudeen/DNN100
File Edit View Search Terminal Help
11:38 PM

```
Epoch 00041: loss improved from 0.00242 to 0.00235, saving model to kddresults/dn1layer/checkpoint-41.hdf5
Epoch 42/50
494021/494021 [==============================] - 62s 125us/step - loss: 0.0023 - accuracy: 0.9993

Epoch 00042: loss improved from 0.00235 to 0.00230, saving model to kddresults/dn1layer/checkpoint-42.hdf5
Epoch 43/50
494021/494021 [==============================] - 60s 122us/step - loss: 0.0023 - accuracy: 0.9993

Epoch 00043: loss did not improve from 0.00230
Epoch 44/50
494021/494021 [==============================] - 63s 128us/step - loss: 0.0022 - accuracy: 0.9992

Epoch 00044: loss improved from 0.00230 to 0.00224, saving model to kddresults/dn1layer/checkpoint-44.hdf5
Epoch 45/50
494021/494021 [==============================] - 61s 124us/step - loss: 0.0022 - accuracy: 0.9993

Epoch 00045: loss improved from 0.00224 to 0.00215, saving model to kddresults/dn1layer/checkpoint-45.hdf5
Epoch 46/50
494021/494021 [==============================] - 60s 122us/step - loss: 0.0022 - accuracy: 0.9993

Epoch 00046: loss did not improve from 0.00215
Epoch 47/50
494021/494021 [==============================] - 61s 124us/step - loss: 0.0021 - accuracy: 0.9993

Epoch 00047: loss improved from 0.00215 to 0.00215, saving model to kddresults/dn1layer/checkpoint-47.hdf5
Epoch 48/50
494021/494021 [==============================] - 70s 141us/step - loss: 0.0021 - accuracy: 0.9993

Epoch 00048: loss improved from 0.00215 to 0.00210, saving model to kddresults/dn1layer/checkpoint-48.hdf5
Epoch 49/50
494021/494021 [==============================] - 77s 156us/step - loss: 0.0022 - accuracy: 0.9993

Epoch 00049: loss did not improve from 0.00210
Epoch 50/50
494021/494021 [==============================] - 89s 180us/step - loss: 0.0021 - accuracy: 0.9993

Epoch 00050: loss improved from 0.00210 to 0.00206, saving model to kddresults/dn1layer/checkpoint-50.hdf5
(base) ict@TLT:~/Downloads/Downloaded Full IDS/Aminudeen/DNN100$
```