International Journal of Progressive Sciences and Technologies (IJPSAT)
ISSN: 2509-0119.

Vol. 42 No. 1 December 2023, pp. 105-113

# *Modified Of Evaluating Shallow And Deep Neural Networks For Network Intrusion Detection Systems In Cyber Security*

Tangang Qisthina Handayani Zatadini, Achmad Farid Wadjdi, I Made Wiryana, Gilang Prakoso, Fadhil Muhammad, Cahya Maharani Badzlina Zataamani, H., Ruby Alamsyah, H.A. Danang Rimbawa

Faculty of Defense Science and Technology, Cyber Defense Engineering
Faculty of Defense Management, Energy Security
Republic of Indonesia Defense University
Jakarta, Indonesia
E-mail: tangangqisthina@gmail.com



**Abstract—Intrusion Detection Systems (IDS) have developed into a crucial layer in all contemporary Information and Communication Technology (ICT) systems as a result of a demand for cyber safety in real-world situations. IDS advises integrating Deep Neural Networks (DNN) because, among other things, it might be challenging to identify certain types of assaults and advanced cyberattacks are complex (DNNs). DNNs were employed in this study to anticipate Network Intrusion Detection System attacks (N-IDS). The network has been trained and benchmarked using the KDDCup-'99 dataset, and a DNN with a learning rate of 0.001 is used, running for 10 epochs for using the activation model experiment and 8 epochs for using the TensorFlow experiment.**

**Keywords—Intrusion detection system, deep neural networks, machine learning, deep learning**

## I. INTRODUCTION

An intrusion-detection system (IDS) is a collection of tools, techniques, and resources that can be applied to detect, evaluate, and report suspicious or illegal network activity. The IDS identifies behavior traffic that may or may not be an intrusion, therefore the "intrusion detection" portion of the term is a bit misleading. It is not a stand-alone security solution; rather, intrusion detection is often a component of an overall security system put around a system or device (Carl Endorf, 2004). IDSs are a variety of cybersecurity- based technologies that were first created to identify flaws and exploits against a target host. The IDS' only function is to find threats. As a result, it is not part of the actual real-time communication channel between the data sender and receiver and is situated outside of the band on the network's infrastructure (Rahul Vigneswaran K).

Implementing the IDS software needs several machine learning algorithms to identify network intrusion. In general, automatic computer processes based on logical or binary operations that learn a task from a set of examples are referred to as machine learning (D. Michie, 1994). In order to identify patterns and add new data to a model, machine learning was created. Without human input, it is capable of error detection and decision-making. The more frequently machine learning is applied and recognizes patterns in the data input, the more accurate a conclusion can be.

Machine learning has several classical algorithms methods, one of which is a decision tree. An illustration of a function that converts a set of attribute values into a single output value, or "decision," is a decision tree. A decision tree makes a choice by running a series of tests starting at the root and moving along the appropriate branch until it reaches a leaf. Each internal node in the tree represents a test of the value of one of the input attributes; the branches from the node are labeled with the potential values of the attribute, and the leaf nodes indicate the result that the function is expected to return (Stuart Russell, 2020).

**Corresponding Author:** Tangang Qisthina Handayani Zatadini

Apart from using decision trees as a classical approach to machine learning, the use of deep learning is also needed as a form of comparison process from machine learning. A wide range of machine learning approaches called "deep learning" use complicated algebraic circuits with configurable connection strengths as their hypotheses. The term "deep" relates to the fact that circuits are frequently layered, meaning that there are numerous steps in the computation paths from inputs to outputs. For applications including visual object recognition, machine translation, speech recognition, speech synthesis, and picture synthesis, deep learning is currently the most popular technique. It also plays a big part in reinforcement learning applications (Stuart Russell, 2020). The use of deep learning is selected using a network structure made up of numerous connected units is known as a neural network (NN). Input, hidden, and output layers make up its three layers of units. The way the units are connected determines the neural network configuration. A network is referred to as a deep neural network when there are more than two hidden layers (DNN) (Selçuk BAYRACI, 2019).

Based on the background that has been described previously, a formulation is formed the main problem, namely how to form a Deep Learning mode using Deep Neural Network modify in the activation model and also in the TenserFlow function. Also, this research purpose is to make modifications to Deep Learning mode using Deep Neural Network with modifications in the activation model and also in the TensorFlow function, where this results in effectiveness in detecting cyber-attacks. This research is limited to the modification of the machine learning model with a decision tree as a form of the classical approach method and deep learning based on deep neural networks to train the KDDCup-'99 data set.

## II. RESEARCH METHODOLOGY

In this research, we are using a true experimental research design: True experimental research relies on statistical analysis to prove or disprove hypotheses, making it the most accurate form of research. Of the types of experimental designs, only true designs can establish cause-and-effect relationships within a group. In a true experiment, three factors must be met: - There is a Control Group, which will not be subject to change, and an Experimental Group, which will be subject to changing variables, a variable that can be manipulated by researchers, and random distribution.

The results of the search that the authors conducted on the literature discussing intrusion detection, the researchers found several previous studies that were relevant to the research the researchers conducted from Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection. This paper presents a plug-and- play NIDS (Network Intrusion Detection System) that can learn to detect attacks on the local network, without supervision, and in an efficient online manner. Kitsune's core algorithm (KitNET) uses an ensemble of neural networks called autoencoders to collectively differentiate between normal and abnormal traffic patterns. KitNET is supported by a feature extraction framework that efficiently tracks the patterns of every network channel. Our evaluations show that Kitsune can detect various attacks with a performance comparable to offline anomaly detectors, even on a Raspberry PI. This demonstrates that Kitsune can be a practical and economic NIDS. The results of the Raspberry PI's benchmark show that a simple network router, with limited resources, can support Kitsune as a NIDS. This means that Kitsune is an inexpensive and reliable distributed NIDS solution. We note that since the experiments were run on a single core, there is potential to increase the packet processing rates further. To achieve this, they plan on parallelizing KitNET over multiple cores. In summation, there is a great benefit in being able to deploy an intelligent NIDS on simple network devices, especially when the entire deployment process is plug-and-play. We hope that Kitsune is beneficial for professionals and researchers alike and that the KitNET algorithm sparks an interest in further developing the domain of online neural network-based anomaly detection (Yisroel Mirsky, 2018).

Second, the paper of Deep Anomaly Detection with Deviation Networks. This paper introduces a novel anomaly detection framework and its instantiation to address these problems. Instead of representation learning, our method fulfils an end-to-end learning of anomaly scores by a neural deviation learning, in which we leverage a few (e.g., multiple to dozens) labeled anomalies and a prior probability to enforce statistically significant deviations of the anomaly scores of anomalies from that of normal data objects in the upper tail. Extensive results show that our method can be trained substantially more data- efficiently and achieves significantly better anomaly scoring than state-of-the-art competing methods. They also find empirically that deep anomaly detectors can be well trained by randomly sampling negative examples from the anomaly- contaminated unlabelled data and positive examples from the small labeled anomaly set. Even when the anomaly contamination level is high, the deep detectors, especially DevNet, can still perform very well and achieve significant improvement over the state-of-the-art unsupervised anomaly detectors. This may provide a new perspective for optimizing anomaly detection methods (Guansong Pang, 2019).

Third, the paper A Taxonomy of Network Threats and the Effect of Current Datasets on Intrusion Detection Systems. This research aims at tackling the problem of having a generic taxonomy for network threats. A proposed taxonomy is presented for categorizing network attacks based on the source, OSI model layer and whether the threat is active or passive. The prominent IDS (Intrusion Detection System) research over the past decade (2008 - 2020) is analyzed. The analysis results in three main findings. First, benchmark datasets lack real-world properties and fail to cope with constant changes in attacks and network architectures, thus, limiting the performance of IDS. Second, we present a taxonomy of tools and associated attacks and demonstrate that current IDS research only covers around 33.3% of threats presented in the taxonomy. Third, we high- light that - whilst ML (Machine Learning) is used by 97.25% of the examined IDS – ANN (Artificial Neural Network), k- means, and SVM (Support Vector Machine) represent the majority of the algorithms used. While these algorithms present outstanding results, we also highlight that these results are obtained on outdated datasets and, therefore, not representative of real-world architectures and attack scenarios. The unique combination of the taxonomy and the analysis of the datasets provided in this manuscript aims to improve the creation of datasets and the collection of real-world data. As a result, this will improve the efficiency of the next-generation IDS and reflect network threats more accurately within new datasets (Hanan Hindy, 2020).

Last, valuating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security the problem discussed Intrusion Detection System (IDS) has become an essential layer in all the latest Information and Communication Technology (ICT) system due to an urge towards Cyber Safety including uncertainty in finding the types of attacks and increased the complexity of advanced Cyber Attacks, IDS calls for the need of integration of Deep Neural Networks (DNNs). DNNs have been utilized to predict the attacks on Network Intrusion Detection System (NIDS). A DNN with 0.1 rate of learning is applied and is run for 1000 number of epochs and KDD Cup-'99' dataset has been used for training and benchmarking the network. For comparison purposes, the training is done on the same dataset with Decision machine learning algorithm and DNN of layers ranging from 1 to 5. The results were compared and concluded that a DNN of 3 layers has superior performance over the Decision Tree machine learning algorithm. IDS is classified into five types, namely NIDS (Network Intrusion Detection System), HIDS (Host Intrusion Detection System), PIDS (Protocol-based Intrusion Detection System), APIDS (Application Protocol-based Intrusion Detection System), and Hybrid (Hybrid Intrusion Detection System). In this paper, we used Network Intrusion Detection System. NIDS is traffic monitoring that is placed at a strategic point covering all hosts in the network. All traffic going to or coming from the network will be analyzed to ascertain whether there are attempted infiltrations in the network system. Because it covers all traffic, it can cause a decrease in network access speed (Rahul Vigneswaran K).

The main objective of this is to analyze and conduct research in ID (Intrusion Detection). A standardized dataset was prepared, which included various types of intrusions which imitated a military environment and was made publicly available. The KDD (Knowledge Discovery in Database) intrusion detection contest's dataset of 1999 was a well-refined version of this. We consider Keras as a wrapper on top of TensorFlow as a software framework. For exponentially increasing the agility of processing of data in deep- learning architectures, a GPU-enabled TensorFlow in multiple devices and Nvidia Video Graphics as follows: the first computer uses the operating system Windows 10 Enterprise 64-bit, Intel Core i7-8750H CPU @ 2.20GHz (12 CPUs), ~2.2Ghz, memory 8192 RAM, GPU NVIDIA GeForce GTX 1050. The second computer uses the operating system Windows 11 Home Single Language 64- bit, AMD Ryzen 3 5300U with Radeon Graphics (8 CPUs) ~2.6GHz, memory 20480 RAM, GPU NVIDIA GeForce GTX 1650 SUPER. The last computer uses the operating system Windows 10 Pro N 64-bit, Intel Xeon CPU X3470 @ 2.9GHz (8 CPUs) ~2.9GHz, memory 8192 RAM, GPU NVIDIA GeForce GT 1030.

## III. Result and Discussion

The type of data used in this research is quantitative data. The DARPA's ID evaluation group, accumulated network- based data of IDS by simulation of an air force base LAN by over 1000s of UNIX nodes and for continuously 9 weeks, 100s of users at a given time in Lincoln Labs which was then divided into 7 and 2 weeks of training and testing respectively to extract the raw dump data TCP. MIT's lab with extensive financial support from DARPA and AFRL, used Windows and UNIX nodes for almost all of the inbound intrusions from an alienated LAN, unlike other OS nodes. For the purpose of the dataset, 7 distinct scenarios and 32 distinct attacks which total up to 300 attacks were simulated.

The data set was used for The Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between bad" connections, called

intrusions or attacks, and good" normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment.

The deep learning model, measurements of each model's activation—the sigmoid, tanh, selu, and softplus—were utilized to assess the segmentation outcomes in this study's evaluation of the findings. To determine the best accuracy using the same dataset from each model activation, each activation is trained for 10 epochs. To make this first experiment, the model activations using same method in define the network's code. Like this table 4.1 in the bottom.

**Table 4.1** Different Model Activation to MakeExperiment Find the Best Accuracy

| Sigmoid | Tanh | Selu | Softplus |
|---|---|---|---|
| model = Sequentia l()<br>model.add(Dense( 1024,inpu<br>t_dim=41,activation<br>='relu')) model.add(<br>Dropout(0<br>.001))<br>model.add(Dense(1)<br>)<br>model.add(<br>Activation('sigmoi<br>d')) | Model = Sequential()<br>model.ad d(Dense( 1024,inp<br>ut_dim=41,<br>activation<br>='relu')) model.add(<br>Dropout(0.001))<br>model.add(Dense(1))<br>model.add( Activatio<br>n('tanh')) | model = Sequential()<br>model.ad d(Dense( 1024,inp<br>ut_dim=41,<br>activation<br>='relu')) model.add(<br>Dropout(0.001))<br>model.add(Dense(1))<br>model.add( Activatio<br>n('selu')) | model = Sequential()<br>model.ad d(Dense( 1024,inpu<br>t_dim=41,activation<br>='relu')) model.ad d(<br>Dropout(0<br>.001))<br>model.ad d(Dense(1<br>))<br>model.ad d( Activatio<br>n('softplu<br>s')) |

After that, we start this experiment to find the best accuracy for each model activation. Then, here the result of this experiment that we see in table 4.2 and the graphic in figure 4.1.

**Table 4.2** The Result of the Experiment

| | Sigmoid | Tanh | Selu | Softplus |
|---|---|---|---|---|
| Epoch 1 | 0,9924 | 0,9899 | 0,991<br>1 | 0,9927 |
| Epoch 2 | 0,9972 | 0,9961 | 0,996<br>5 | 0,9968 |
| Epoch 3 | 0,9975 | 0,9961 | 0,995<br>8 | 0,9966 |
| Epoch 4 | 0,9977 | 0,9966 | 0,996<br>2 | 0,9973 |
| Epoch 5 | 0,9980 | 0,9968 | 0,996<br>9 | 0,9975 |
| Epoch 6 | 0,9982 | 0,9971 | 0,996<br>4 | 0,9978 |
| Epoch 7 | 0,9983 | 0,9972 | 0,997<br>2 | 0,9979 |
| Epoch 8 | 0,9984 | 0,997 | 0,994 | 0,9981 |

| | | | 2 | |
|---|---|---|---|---|
| Epoch 9 | 0,9986 | 0,997 | 0,994 6 | 0,9982 |
| Epoch 10 | 0,9987 | 0,9969 | 0,993 9 | 0,9975 |

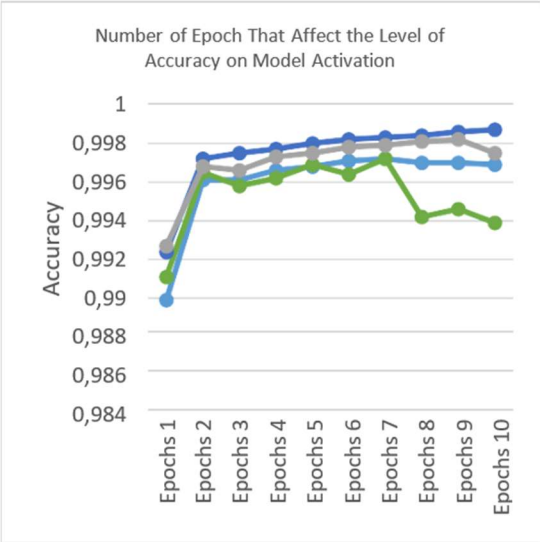


**Figure 4.1** Graphic of the Experiment

As we can see from the table 4.2 also from the figure 4.1, the sigmoid model activation more stable accuracy and increase from each epoch. But in the other hand selu, tanh, and softplus model activation, we can see the accuracy from epoch 2 until epoch 10 unstable. Next, try the second experiment with the model activations using same method in define the network's code. Like this table 4.3 in the bottom.

**Table 4.3** Different Model Activation to MakeExperiment Find the Best Accuracy Second Experiment

| Sigmoid | Tanh | Selu | Softplus |
|---|---|---|---|
| model = Sequentia l()<br>model.add(Dense( 1024,inpu t_dim=41,activation ='relu')) model.add( Dropout(0 .001))<br>model.add(Dense( 512,input _dim=41 ,activation ='relu')) model.add( Dropout(0 .001))<br>model.add(Dense(1) )<br>model.add( Activatio n('sigmoid')) | Model = Sequential()<br>model.ad d(Dense( 1024,inp ut_dim=41, activation ='relu')) model.add(Dense( 512,input _dim=41 ,activation='relu'))model.ad d( Dropout(0.001))<br>model.add( Dropout( 0.001))<br>model.add(Dense(1))<br>model.add( Activatio n('tanh')) | model = Sequential()<br>model.ad d(Dense( 1024,inp ut_dim=41, activation ='relu')) model.add(Dense( 512,input _dim=41 ,activation='relu'))model.ad d( Dropout(0.001))<br>model.add( Dropout( 0.001))<br>model.add(Dense(1))<br>model.add( Activatio n('selu')) | model = Sequential()<br>model.ad d(Dense( 1024,input_dim=41, activation ='relu')) model.ad d( Dropout(0 .001))<br>model.add(Dense(512,input _dim=41 ,activatio n='relu')) model.ad d( Dropout(0 .001))<br>model.ad d(Dense(1 ))<br>model.ad d( Activatio n('softplu s')) |

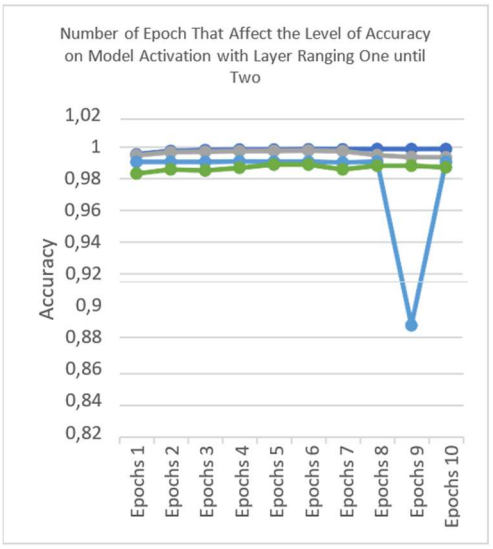


**Figure 4.2** Graphic of the Second Experiment

Also, we can see from the table 4.3 also from the figure 4.3, the sigmoid model activation more stable accuracy and increase from each epoch. But in the other hand selu, tanh, and softplus model activation, we can see the accuracy from epoch 2 until epoch 10 unstable. But this experiment has increase from the first experiment. This table and graphic say sigmoid model activation that Rahul Vigneswaran K has the best accuracy from others model activation. Its because the sigmoid model activation has a function sigmoid(x) = 1 / (1 + exp(-x)). It applies for small values (<-5), sigmoid returns a value close to zero, and for large values (>5) the result of the function gets close to 1.

Next experiment we Modifying and Improvement DNN using TensorFlow. First we used the original source codes, then running it and we got all the results of DNN3 (dnn3.py and dnn3test.py).

Results from dnn3.py, can see how all the 8 epochs from KDD-Cup 99 dataset running and getting all the results for estimated completion time (ETA), loss, accuracy.

We have the best data from loss = 0.00417 (0.0042) and it keep improved. Also we got the best data from accuracy = 0.9989 and it kept going near increasing to have greatest accuracy.

To get that we have the loss kept reduced also increased results from the accuracy, its very good already and concluded that a DNN of 3 layers has superior performance over all the other classical machine learning algorithms.

Results from dnn3test.py, Can see that the results for accuracy, recall, precision, f1score datas were very great. We have accuracy = 0.924, recall = 0.907, precision = 0.999, f1score = 0.951 for the DNN3test.

Now here we want to make modify the DNN and improvement DNNtest using TensorFlow, because we're curious by adding these codes.

```
import tensorflow as tf
import tensorflow_datasets as tfds

# Construct a tf.data.Dataset
ds = tfds.load('mnist', split='train',
shuffle_files=True)

# Build your input pipeline
ds =
ds.shuffle(1024).batch(32).prefetch(tf.data.
AUTOTUNE)
for example in ds.take(1):
  image, label = example["image"],
example["label"]
```

dnn3.py results, We can see from the results NIDS and getting all the results for estimated completion time (ETA), loss, accuracy. The epochs worked more burdened finding the types of attacks and increased the complexity of advanced Cyber Attacks.

After modify it, we have the best data from loss = 0.00443 (0.0044) and it keep improved. Also we got the best data from accuracy = 0.9987.

dnn3test.py results, We can see that the results for accuracy, recall, precision, f1score datas after used TensorFlow increasing. We have accuracy = 0.927, recall = 0.911, precision = 0.998, f1score = 0.953 for the DNN3test.

We can say that, when the DNN (Deep Learning) modified using TensorFlow the results have improvement. See this datas comparison:

**Table 4.4** From orginal souce codes to Modifying and Improvement DNN using TensorFlow results (dnn3.py and dnn3test.py)

| | Dnn3 (original sourcer) | Dnn3test (original source) | Dnn3 (modifying and improvement with TensorFlow) | Dnn3test (modifying and improvement with TensorFlow) |
|---|---|---|---|---|
| Epoch 1 | loss = 0,0130<br>accuracy = 0.9958 | loss = 0.0129<br>accuracy = 0.9959 | loss = 0.0130<br>accuracy = 0.9959 | loss = 0.0130<br>accuracy = 0.9958 |
| Epoch 2 | loss = 0,0075<br>accuracy = 0.9978 | loss = 0.0073<br>accuracy = 0.9978 | loss = 0.0079<br>accuracy = 0.9977 | loss = 0.0076<br>accuracy = 0.9978 |
| Epoch 3 | loss = 0,0061<br>accuracy = 0.9982 | loss = 0.0065<br>accuracy = 0.9981 | loss = 0.0067<br>accuracy = 0.9983 | loss = 0.0064<br>accuracy = 0.9982 |
| Epoch 4 | loss = 0,0056<br>accuracy = 0.9984 | loss = 0.0056<br>accuracy = 0.9984 | loss = 0.0064<br>accuracy = 0.9984 | loss = 0.0056<br>accuracy =0.9984 |
| Epoch 5 | loss = 0,0050<br>accuracy = 0.9985 | loss = 0.0050<br>accuracy = 0.9986 | loss = 0.0053<br>accuracy = 0.9985 | loss = 0.0053<br>accuracy = 0.9985 |
| Epoch 6 | loss = 0,0046<br>accuracy = 0.9987 | loss = 0.0048<br>accuracy = 0.9986 | loss = 0.0050<br>accuracy = 0.9986 | loss = 0.0047<br>accuracy = 0.9986 |
| Epoch 7 | loss = 0,0042<br>accuracy = 0.9988 | loss = 0.0045<br>accuracy = 0.9987 | loss = 0.0048<br>accuracy = 0.9987 | loss = 0.0045<br>accuracy = 0.9987 |
| Epoch 8 | loss = 0,0042<br>accuracy = 0.9989 | loss = 0.0044<br>accuracy = 0.9988 | loss = 0.0044<br>accuracy = 0.9987 | loss = 0.0043<br>accuracy = 0.9988 |

**Table 4.5** DNN3 results Before with original sourcecodes (left) and the results After modify and improvement using TensorFlow (right)

| Dnn3test (original sourcer) | Dnn3test (modifying and improvement with TensorFlow) |
|---|---|
| accuracy = 0.924<br>recall = 0.907<br>precision = 0.999<br>f1score = 0.951 | accuracy = 0.927<br>recall = 0.911<br>precision = 0.998<br>f1score = 0.953 |

## IV. CONCLUSION

We can see our improvement in DNN using tensorflow from the result for estimated completion time (ETA), loss, accuracy from 0.924 become 0.927, following by recall from 0.907 to 0.911, talking about total score which before improvement is 0.951 become 0.953. and from the other side, We find the deficiency from the classical approcahes on Decision Tree model (Machine Learning). When the Classical Approaches file Decision Tree (Machine Learning) is created its own file and the command is separated from other ML models there is a change in the results of Accuracy score, Precision score, Recall score, F1score or all the parameters. From accuracy from 0.928 become 0.930, continuing with racall from 0.912 is increase become 0.914 nad there is also the results of classical machine learning Decision Tree not stable.

## ACKNOWLEDGMENT

The authors really appreciated and would like to thank all those who have helped and supported this research. Special thanks to the Republic of Indonesia Defense University and the entire academic community.

## REFERENCES

[1] Adhitya Prananda, Yusuf, & Rudy A.G. Gultom. (2021). Sinergi Lembaga Intelijen dalam Menghadapi Ancaman Siber di Indonesia. Jurnal Peperangan Asimetris, Universitas Pertahanan Republik Indonesia, Vol 7(1), 51–70.

[2] Carl Endorf, E. S. (2004). Intrusion Detection & Prevention. California, U.S.A: Brandon A. Nordin.

[3] Cybersecurity Best Practices. Retrieved from https://www.cisa.gov/topics/cybersecurity-best-practices

[4] D. Michie, D. S. (1994). Machine Learning, Neural, and Statistical Classification.

[5] Guansong Pang, C. S. (2019). Deep Anomaly Detection with Deviation Networks.

[6] Hanan Hindy, D. B. (2020). A Taxonomy of Network Threats and the Effect of Current Datasets on Intrusion Detection Systems.

[7] Indonesia responds to the cyber dark side. Retrieved from https://www.lowyinstitute.org/the-interpreter/indonesia-responds-cyber-dark-side

[8] keras.io. (n.d.). Layer activation functions. Retrieved from keras.io: https://keras.io/api/layers/activations/

[9] Luke Irwin. 8th September 2022. The 5 Stages of an Effective Cyber Defence Strategy. Retrived from https://www.itgovernance.eu/blog/en/the-5-stages-of-an-effective-cyber-defence-strategy

[10] N. Tran, H. Chen, J. Bhuyan and J. Ding. (October, 2022). "Data Curation and Quality Evaluation for Machine Learning-Based Cyber Intrusion Detection," in IEEE Access, vol. 10, pp. 121900-121923, 2022, https://ieeexplore.ieee.org/document/9907008

[11] Peraturan Presiden (PERPRES) No. 47 Tahun 2023 tentang Strategi Keamanan Siber Nasional dan Manajemen Krisis Siber (Presidential Regulation (PERPRES) No. 47 of 2023 concerning National Cybersecurity Strategy and Cyber Crisis Management). https://peraturan.bpk.go.id/Details/255542/perpres-no-47-tahun-2023. https://drive.google.com/file/d/1v6kAQTXE4VtJY9KtWtDDuMnjYry8jxBs/view

[12] Rahul Vigneswaran K, V. R. (n.d.). Evaluating Shallow and Deep Neural Networks for Network Intrusion Detection Systems in Cyber Security.

[13] Selçuk BAYRACI, O. S. (2019). A Deep Neural Network (DNN) based classification model in application to loan default prediction. Theoretical and Applied Economics, 75-84.

[14] Stuart Russell, P. N. (2020). Artificial Intelligence A Modern Apporach Fourth Edition. Pearson.

[15] Yisroel Mirsky, T. D. (2018). Kitsune: An Ensemble of Autoencoders for Online Network Intrusion Detection.