Available online at https://asric.africa/natural-sciences
ASRIC Journal on Natural Sciences Vol 3(1) (2023) 182-193

Ctgan Adversarial Attack on Network Intrusion Detection Based on Lstm Algorithm

Ahmad Abubakar Yunusa[1], Fatima Umar. Zambuk[2], Badamasi Imam. Ya'u[3], Abubakar Umar[4], Abdulkadir Hassan Disina[5]

[1]Computer Science Department,
Nigerian Army University Biu, Borno, Nigeria.
[2, 3, 4] Department of Mathematical Sciences,
Abubakar Tafawa Balewa University, Bauchi, Nigeria.
[5]Cyber Security Department,
Nigerian Army University Biu, Borno, Nigeria.

*Corresponding author: Ahmad Abubakar Yunusa

Abstract
Deep neural networks have proven successful in the intrusion detection domain. Cyber security experts and designers must develop a variety of network intrusion detection systems to secure networks and computers from black hackers who might breach the network system and steal or damage important data from databases. Regrettably, recent studies revealed that adversarial samples can affect deep neural networks. Since it is commonly known that deep learning algorithms are susceptible to adversarial examples, cybercriminals dare to devise ways to utilize weaknesses to acquire data or try to mislead these intrusion detection classifiers to make them misclassify data for their selfish advantage. But since there are too many parameters, Deep Neural Networks (DNN) show structural instability, which reduces the model's overall accuracy. Based on the literature on adversarial samples creation for neural networks, the priority was in models that dealt with classification problems. Consequently, it does not address intrusion detection based on time series data. In this research, we study the performance of the CTGAN attack method against an LSTM-based method for detecting network intrusions. We initially use the NSL-KDD dataset to train an LSTM model for detecting malicious behavior and our results show that the classifier has achieved good performance with accuracy of 0.9607, Precision 0.8725, Recall 0.3873, and F1 score 0.9210, then we generate adversarial data using CTGAN synthesizer on the dataset there by making the LSTM model misclassify an attack as normal record and the attack's success rate was evaluated making the results with accuracy of 0.5257, Precision 0.5656, Recall 1.0000, and F1 score 0.4099.

Keywords: Machine Learning; Deep Learning; Network Intrusion Detection Systems; Long-Short-Term Memory; Adversarial Examples; Conditional Tabular Generative Adversarial Network.

I. Introduction
A network is a congregation of Computers, Servers, Mainframes, and network devices like switches, routers, network terminal access points, Peripherals, or Other devices to easily share data. Since the internet is an interrelationship of a few networks, it has been observed that it plays a significant role in our day-to-day activities through frequent use of the internet (Dua and Du, 2016). This gave cyber criminals the courage to devise a means of extracting data and implementing vulnerabilities by any means possible for their selfish consumption, Consequently, To safeguard networks and computers from hackers who might attack the systems and take or ruin medical, financial, or other sensitive information from databases, cyber security experts and designers must create a variety of intrusion detection systems (Dua and Du, 2016). With this, several ways of protecting these data were developed by security experts to save victims from those vulnerabilities, as there are a lot of machine learning algorithms such as Hidden Markov Models,

Fuzzy Logic, (Jha et al., 2001) and Neural Networks (Kaur & Gill, 2013; Wu and Banzhaf, 2010) have made great achievements in an intrusion detection system. The majority of conventional machine learning techniques with shallow architectures either contain only one hidden layer (for example, ANN) or none at all (for example, Maximum Entropy Model) (Gao et al., 2015).
Intrusion Detection System (IDS) is an important research development in the area of information security that can identify an intrusion, it could be an intrusion that is happening today now or one that has already happened. Typically, intrusion detection relates to a classification issue, like a binary or multiclass classification algorithm., that determines if network traffic conditions are anomalous, normal, or a four-category classification, identifying whether it is normal or any one of Probe, User to Root (U2R), Denial of Service (DOS), Remote to Local (R2L), attack types. In short, improving the classifier's intrusion detection accuracy is the major motivation for intrusion detection. (Yin et al., 2017).

A network or host system can use the IDS to quickly detect intrusions, attacks, or security policy breaches. It is essential for protecting against unauthorized access and malicious activity. Based on operational logic, there are two basic categories of IDS: Both signature-based which compares a database of known threats, and anomaly-based IDS which inspects communications based on the behavior of activities. (Wang, 2018). Many intrusion detection systems make use of Machine Learning (ML) Techniques in deploying security measures to computerized information to provide good discretion on the information. Network Intrusion Detection System is the name of an IDS system that examines a packet of networks to find intrusions (NIDS). A NIDS is obtained by mirroring the aforementioned network devices. Deep learning is acknowledged as a suitable approach in NIDS, however recent research has shown that these models are susceptible to intentionally generated inputs known as adversarial examples (Szegedy et al., 2014).

Adversarial examples are created by introducing minor but purposefully chosen perturbations to the original data that caused the records to be inappropriately identified by the deep learning models. These Adversarial examples include the Fast gradient sign method (FGSM) (Goodfellow et al., 2015), Jacobian-based saliency map attack (JSMA) (Papernot et al., 2016), DeepFool (Moosavi-Dezfooli et al., 2016) and C&W attack (Carlini and Wagner, 2017).

Yang et al., (2019) developed a deep neural network (DNN) model for NIDS, an attempt was made to use adversarial samples to cause the model to misclassify. Three alternative attack strategies were examined in a black-box model to create adversary examples. These attack strategies e.g. C&W, ZOO, and GAN were used in launching adversarial attacks.

In this paper, we investigate the success rate of network intrusion detection based on an LSTM algorithm using NSL-KDD dataset and the likelihood of launching CTGAN adversarial attack method on the NIDS.

II. Related Works

Rigaki and Elragal, (2017) evaluated the impact of adversarial examples in a scenario involving intrusion detection. The NSL-KDD dataset was used by the authors for testing, generating Targeted attacks with FGSM and JSMA. Random forest, decision tree, linear SVM, multi-layer perceptron (MLP), neural network, and voting ensembles of the earlier three classification models to perform classification. The results showed that linear SVM is the most classifier with a decrease of 27% on JSMA. Random forest was the most reliable classifier, with reduced performance drops across all parameters, including an 18% drop in accuracy and a 6% drop in F1 score and AUC. Also, it was highlighted that while the investigated attacks verified their effectiveness against unidentified classifiers, the success of the cyber-attacks under different conditions has not been shown, and they still required an understanding of the data preprocessing, such as the classification features.

Wang, (2018) used the Multi-Layer Perceptron classifier in this research to assess how well modern attack methods perform using the NSL-KDD dataset for intrusion detection based on a deep learning algorithm. The performance of the MLP neural network was Evaluated, Attacks by CW tend to be less harmful than those by JSMA., FGSM, and Deepfool attacks had an AUC of 0.80, whereas the targeting FGSM had an AUC of 0.44 and was the most successful attack, In terms of usability and pooling at the initial stage, JSMA

attacks are considerably more alluring for an adversary, generating an AUC of about 0.5 across all classes, however, because the size and parameters are increased by the LSTM cell, LSTM-CNN takes longer to train. Although the DNN is rather fast, it has low accuracy and specificity scores. Also, the authors examined various attacks on the improved features, discovering that almost all attacks use the same seven features, as well as describing how an attacker might physically edit each one to undertake adversarial attacks in a realistic scenario. The attacker is conscious of the target's deep neural networks, and white-box attacks were taken into consideration.

Lin et al., (2018) Adversarial techniques that can fool and defeat the intrusion detection system were generated using a variant of a WGAN dubbed IDSGAN on the NSL-KDD dataset. The authors created adversarial network traffic by using a generator to construct adversarial attack traffic on real malicious traffic and a discriminator to distinguish between the two. The classifiers tested for the effectiveness of the threats were naive Bayes, SVM, MLP, neural network, decision tree, k-nearest neighbors, logistic regression, and random forest. IDSGAN takes advantage of a generator to turn actual harmful data into adversarial malicious activity and a discriminator to identify examples of legitimate traffic from unwanted traffic. Adversarial attack examples carry out black-box attacks against the detection system. The findings demonstrated a reduction in adversarial example identification rates across all classifiers. The authors concluded that the suggested technique works well at producing adversarial attacks by producing fresh malicious samples. More research can propose to test this strategy on more attack types.

Yan et al., (2019) Introduced DoS-WGAN, a typical structure that takes utilization of Wasserstein generative adversarial networks (WGAN) and gradient penalty technology to disguise offensive denial of service (DoS) attacks as regular network traffic to get against network traffic classifiers, which the NSL-KDD uses. Four categories have been developed by the authors to group the features: i) Basic features which refer to the connection's most fundamental characteristics ii) Content features, which correspond to the connection's content and are unimportant for DoS attacks iii) the traffic features across a two-second timeframe are the resulting output of all interactions made in the past two seconds makeup; iv) Statistics for the 100 prior connections are shown in the traffic features over 100 connections window. To train the WGAN, adversarial DoS samples were created and approximated to the distribution of ordinary traffic inputs, between the generator and the discriminator, a converter was added to prevent the modification of the content features and traffic features of 100 connection units. The true positive rate of a CNN classifier was reduced, demonstrating the impact of the attack can be increased while keeping its integrity by not altering the denial of service attack's functioning features. Conclusively, DoS-WGAN is more stable and more efficient than other methods for creating and producing samples in opposition to a CNN-based IDS.

Yang et al., (2019) investigate how adversarial instances affect the effectiveness of the deep neural network (DNN) developed to identify anomalous activities within black-box adversarial methods. The DNN classifier was trained to detect NIDS with three attack techniques on the NSL-KDD dataset, these attack strategies were C&W, GAN, and Zoo. In the beginning, to attack a different trained model using C&W, another model was built and utilized to create adversarial samples. In the second instance, without first training a replacement, attack the target classifier, by querying the classifier, ZOO was utilized to calculate the gradient and produce adversarial attacks. In generating adversarial samples, WGAN was trained in the third scenario. Attack based on ZOO outperforms the Substitute model the most effective but least realistic strategy is GAN because it took too many queries to get adversarial samples that would be seen in a real setting.

Martins et al., (2019) evaluated the effect of JSMA FGSM, C&W, and Deepfool adversarial attacks on the CICIDS2017 and NSL-KDD datasets. Existing malicious samples were modified to analyze the footprint of multiple classifiers which include: random forest, decision tree, naive Bayes, SVM, DAE, and neural network. The DAE was utilized to classify images by initially training the autoencoder, freezing the layers, and then training additional layers connected to the autoencoder's output, which performed classification. The performance of the classifiers was shown to be deteriorated by all classifiers employing an average AUC, the NSL-KDD dataset and the CICIDS dataset saw decreases of 13% and 40%, respectively. The

DAE was found to be the most resilient classifier. Adversarial training was then applied to all classifiers as a defensive measure, it was shown that all classifiers performed better, with the random forest on average outperforming the others on the two datasets.

Apruzzese et al., (2019) carried out attacks on the network intrusion detectors' integrity. On the CTU-13 dataset, MLP neural networks and KNN models were used. Having divided the detectors into numerous instances, each dedicated to a different botnet family, Random forest was discovered to be the best-performing detector, whereas KNN was the worst. A custom adversarial attack was then implemented targeting three attributes: *exchanged_bytes* (amount of data shared), *duration* (time spent connecting), and *total_packets* (amount of packets exchanged. This approach operates by adding random values to each feature over a predetermined interval, resulting in new adversarial samples. Random forest is found to be the best-performing classifier, whereas MLP neural network is the worst. Adversarial training and feature removal defenses were then tested where the recall was increased for both Random forest and KNN While applying feature removal, the recall improved for both MLP and random forest when using adversarial training.

Apruzzese and Colajanni, (2018) by changing network flows alone utilizing the CTU13 dataset, the authors investigated the efficacy of adversarial examples performed in a situation when an attacker attempts to covertly spread botnet software throughout a large network while remaining undiscovered. A random forest detector trained on active botnets utilizing network flow parameters like connection protocol, IP addresses, ports, and packets sent was used on the dataset. The dataset contains 7 different varieties of malware. 7 classifiers (1 for each attack type) used training sets with 95% normal and 5% harmful samples for training. With two exceptions, the detection rate was generally greater than 0.95; one of these should be highlighted because of how little sample data was provided for the Sugou attack, which was one of the outliers. The number of packets, period of the flow, bytes exchanged, and up to four other parameters of the original malicious flows are all randomly altered. The detection rates for each type of attack were drastically reduced, according to the number of variables altered and the amount of the modification, decreasing detection performance.

Huang et al., (2020) evaluated the efficiency of adversarial samples by generating DDoS adversarial samples using two methods: Probability Weighted Packet Saliency Attack (PWPSA) and Genetic Attack (GA) on the CICIDS2017 dataset. Both techniques modify the initial input sample by adding or removing missing packets. In GA, the authors use the genetic algorithm to create a group of modified samples and then find the best variation among them. In PWPSA, the authors iteratively change the original sample, using position saliency and at each stage, the insertion or replacement order is determined by the packet score. The Experiments demonstrate that both strategies have a high success rate in bypassing DDoS detectors therefore both GA and PWPSA methods are powerful and effective.

III. Background

A. Network Intrusion Detection System (NIDS)

Intrusion Detection Systems (IDS) are computer applications that are designed to detect abnormal or harmful attempts to access the computer network (Lakshminarayana et al., 2019). They are a crucial strategy in any policy to address cyber security issues. An intrusion detection system (IDS) enables one to discover whether a network is being attacked (Alsaeedi and Khan, 2019). Once an alert is received that a network is attacked, an action to quickly prevent that attack from taking over the system is taken. Machine Learning and Deep Learning techniques. These intrusion detection systems employ supervised, unsupervised, and semi-supervised techniques (Heng and Weise, 2019). Gao et al., (2015) proposed a deep belief network-based intrusion detection model, and khorram (2021) built an IDS model based on the most effective Machine learning algorithms i.e. KNN, SVM, and RF algorithms.

A NIDS is a system that identifies harmful behavior in a network (Chandran and Kumar, 2018). Typical abnormal network activities include Denial of service (DoS), remote to local (R2L), user to root (U2R), probe, etc are common malicious network actions (Yang et al., 2019).

B. Long Short-Term Memory (LSTM)

A version of RNN called LSTM neural networks uses special units alongside conventional units (Shende, 2020). These LSTM units comprise a 'memory cell' which can store records in the memory for a long time. There are three different kinds of gates: input, output, and forget. A collection of gates is utilized to regulate the entry of data into the memory, forgetting, and output layers (Aggarwal, 2018). The input gate determines how much data from the previous sample will be stored in memory; the output gate controls how much data is sent to the next layer; while forget gates manage the rate at which memory is torn out of storage. They can discover longer-term dependencies due to this architecture (Laghrissi et al., 2021).

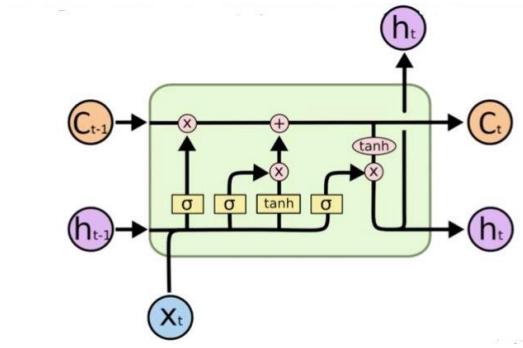

***Figure 1**: LSTM Architecture Source: (Shende, 2020)*

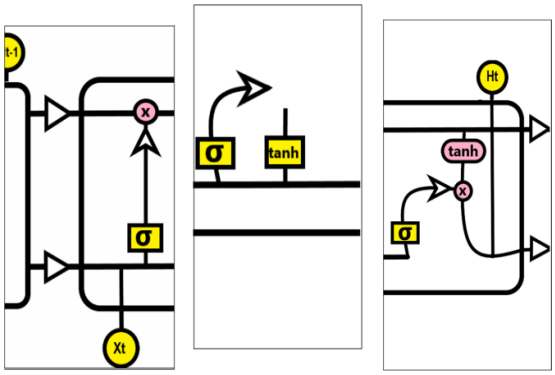

***Figure 2:** a. Forget Gate Layer b. Update /Input Gate Layer c. Output Gate Layer (Laghrissi et al., 2021)*

C. Adversarial Examples

Almost every element of our everyday life is now impacted by an information technology infrastructure. However, critical services and user data are now more exposed to cyber-attacks due to this level of seamless connection (Goodfellow et al., 2015). Adversarial samples are particular inputs that are intended to perplex a neural network and make it misclassify an input (Szegedy et al., 2014). Although these well-known inputs are invisible to the human eye, they prohibit the network from accurately determining the image's contents. As threats become more complex and prevalent, researchers and security experts must develop innovative knowledge, in-depth understanding, and the appropriate abilities to address these issues (Heng & Weise, 2019). Hence, Threat modeling, Attack simulation, counter major design noise detection (for evasion-based attacks), and Information laundering were the approaches to protecting machine learning introduced by researchers. Application scenarios of these Adversarial attack strategies are either black-box attacks or white-box attacks. White-box attacks provide attackers to gain access to the classifier's training data, model parameters, and other important details. While in a "black-box" case, the intruder has little to no information about the classifier and is therefore greatly constrained (Martins et al., 2020).

There are several types of such attacks strategies, which include: Jacobian-based Saliency Map Attack (JSMA), Limited-memory Broyden Fletcher Goldfarb Shanno (L-BFGS), Fast Gradient Sign Method (FGSM), Deepfool, Carlini and Wagner (C&W), Zeroth Order Optimization (ZOO), Generative adversarial networks (GAN), etc.

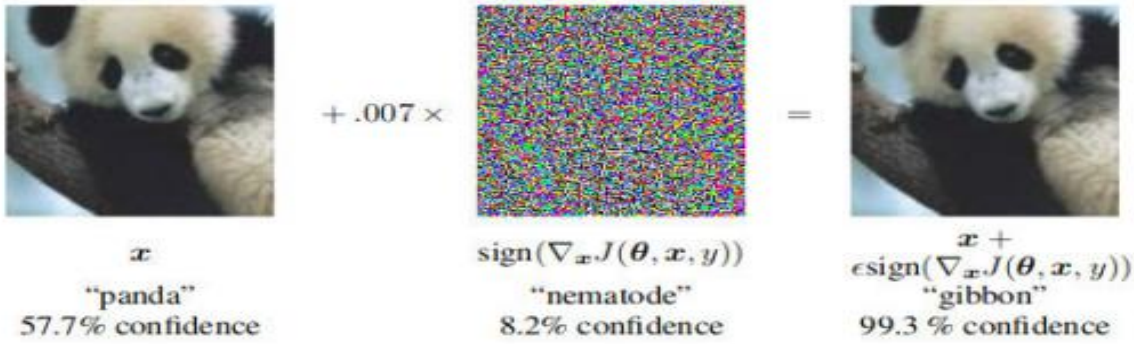

Figure 3: *Demonstration of an Adversarial sample generated by applying FGSM to GoogleNet. Left: Panda Image, Middle: Perturbed, and Right: Gibbon (Adversarial Image).* (Ren et al., 2020)

IV. Evaluation Methodology

A. Methodology

This research work investigates the effects of adversarial actions affecting the performances of a cyber-intrusion detection system built based on Long Short-Term Memory (LSTM) that is trained to identify anomalous behavior in the black-box model. The proposed LSTM model will perform binary classification on NSL-KDD dataset training and test sets to identify harmful activities in a network considering performance, CTGAN adversary attack will be tried to launch a black-box attack by generating fake samples to the original input thereby making the records' maliciousness to cause the LSTM model to misclassify. Figure 4 illustrates the framework of the proposed system. Primarily, the first step of the framework involves the data set and a preprocessing layer, next stage is the training of the LSTM classifier for intrusion detection, then employing CTGAN strategy for generating fake data, re-training of the classifier on the same setup, and finally evaluate the performance based on the accuracy, precision, recall, and F1 score evaluation matrices.

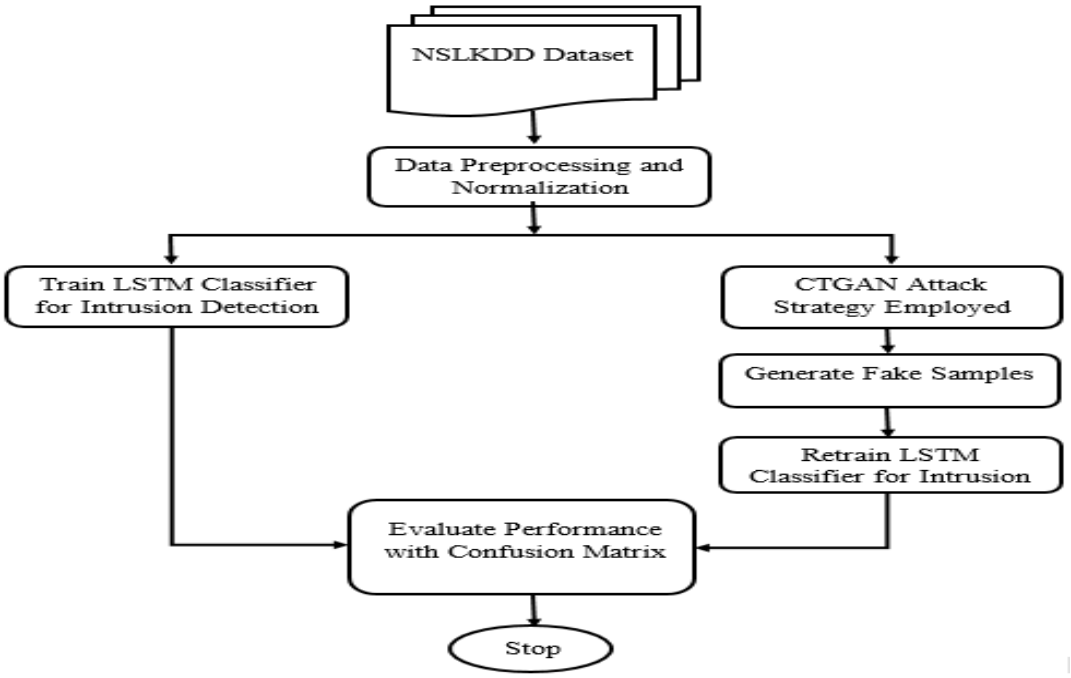

Figure 4*: The proposed System*

B. Dataset Description

The NSL-KD dataset was collected from the Kaggle website which contains all the codes and data needed to carry out data science projects. It is an open source on the organization's official website (https://www.kaggle.com/hassan06/nslkdd). The records available in the NSL-KDD dataset cover four types of attacks: DoS, R2L, probing, and U2R attacks are all attacks, with the remaining ones denoting regular traffic flow. The NSL-KDD dataset consists of three different types of data: binary, continuous, and symbolic.

Table 1: Records available in the NSL-KDD Dataset (Yang et al., 2019)

Traffic Label	No. of Training Records	No. of Test Records
DOS	45927	7548
U2R	52	67
R2L	995	2887
Probe	11656	2421
Normal	67343	9711

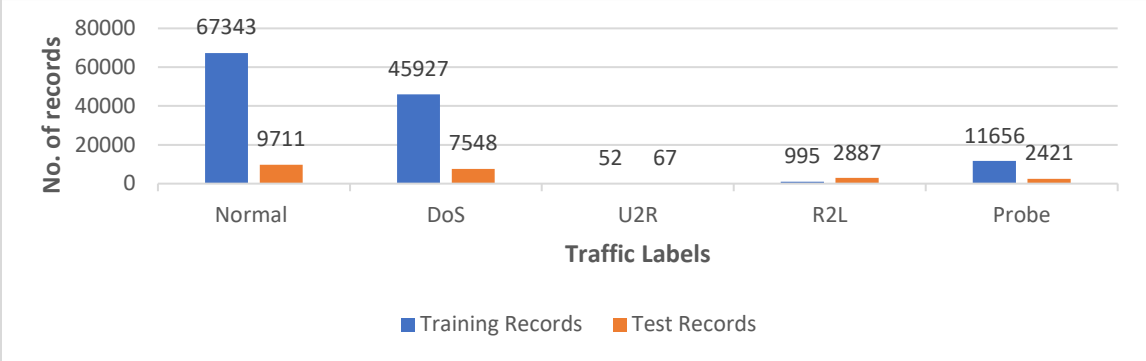

Figure 5: *Number of records for each attack type in the NSL-KDD dataset.*

The Dataset's records need to be preprocessed before training. Both symbolic and binary parameters must be converted to numbers using one-hot encoding. Also, we transformed the 22 attack labels from the dataset to the normal, R2l, U2R, Probe, and DoS traffic flows attack classes. We also appled standard scalar to normalize the futures to transform the data so that we can have a standard deviation of 1 and its distribution will have a mean value of 0.

C. Training the LSTM for Intrusion Detection

Based on our proposal to achieve an intrusion detection classifier first, a deep learning LSTM method will be applied for network intrusion detection. This method has several steps. The Model will specifically apply one input layer with 42 features of the NSL-KDD dataset's records after preprocessing during the training procedure as model's inputs x. and one output classification scheme which will use binary classification to distinguish between normal and abnormal data for testing where 0 will be set as abnormal and 1 will be set as normal data with 75% for training and 25% for testing on the dataset.

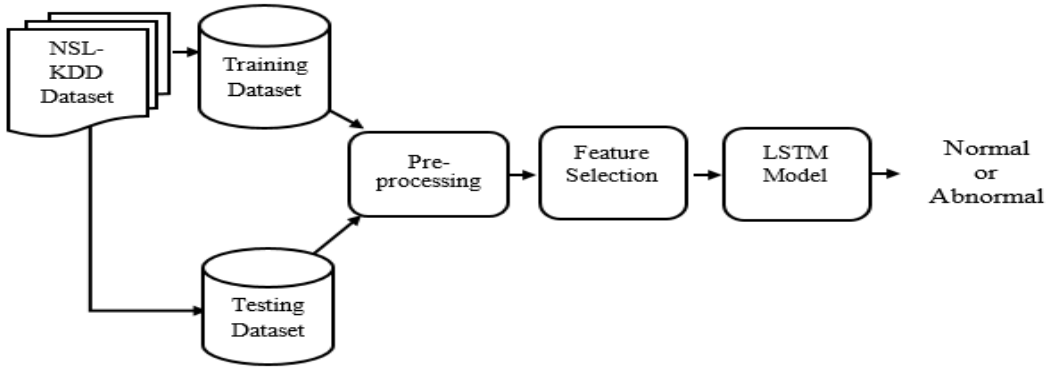

Figure 6: *The Proposed LSTM Model*

D. CTGAN-based attack technique

A black-box attack based on a Conditional Tabular Generative Adversarial Network will be launched by the adversary. The dataset is divided into 70% for training and 30% for testing. First, we would create fake input variables X_{fake} and train a CTGAN using input variables X_{real}. The CTGAN can produce variables that are comparable to X_{real} as it gains knowledge of $X_{real's}$ attributes through this training. The training process involves simulating records one by one.

The generator's output dimensions would be modified to account for the dataset's form to feed the model for the evaluation of fake synthetic samples.

E. Performance Evaluation Metrics

The following metrics can be used to assess how well this classifier performs: accuracy (AC), false alarm (FA), precision (PC), and Fscore.

i. Accuracy: this performance metric deals with the correct prediction made by the model and this evaluation metric is expressed as:

$$AC = \frac{TP+TN}{TP+FP+FN+TN} \quad (1)$$

ii. Precision: Precisions show you how correct and precise your model is based on those expected positives and the number of them positives. When the cost of false positives is high, precision is a good indicator. It is written as:

$$PC = \frac{TP}{TP+FP} \quad (2)$$

iii. Recall (RC) is measured as the proportion of malicious records that have been correctly identified overall all attack records and it is evaluated by:

$$RC = \frac{TP}{TP+FN} \quad (3)$$

iv. FA is the ratio of the number of normal records classified as malicious to the total number of normal records. it is given by:

$$FA = \frac{FP}{FP+TN} \quad (4)$$

v. Fscore is given by the harmonic mean of precision and recall and is determined as a derived measure of efficacy, it is evaluated by:

$$Fscore = 2.\frac{PC.RC}{PC+RC} \quad (5)$$

Where TP (True Positive) = Are instances where the expected class matches the data item's actual class, which was 1 (true).

TN (True Negative) = Are instances where both the expected and actual classes of a data point are 0 (false).
FP (False Positive) = Are instances where the expected class is 1 (true) but the real category of the data item was 0 (false).

FN (False Negative) = Are there instances where the expected class is 0 (false) but the real category of the data item was 1 (true)?

V. Experiments and Analysis

The data in the NSL-KDD dataset are used to train an LSTM model for binary classification, and its success is assessed using a test set for network intrusion detection. Then, to target this model, we created records using the black-box attack strategies described earlier. We test the viability of the generated adversarial attacks in the NIDS sector through experimentation and analysis. All experiments were conducted on an Intel Core i7 laptop running Windows 10 operating system. Tensorflow, Numpy, Pandas, Keras, Sklearn, and Matplotlib libraries were used.

A. Performance of the LSTM Model for intrusion detection

A binary LSTM model was built with 3 input LSTM layers containing 100 units each, 1 output dense layer with a sigmoid activation function, and an Adam optimizer to calculate the accuracy of the intrusion detection classifier successfully based on the NSL-KDD training and test sets.

Confusion metrics were plotted to assess the LSTM algorithm's performance. The occurrences in each column of the metrics correspond to an actual class, whereas the cases in each row of the metrics relate to a predicted class. Therefore, the true positives, false positives, true negatives, and false negatives were measured using table metrics described in 3.7 of section three. It can be observed from Table 2 that our model (LSTM Classifier) shows good performance during training.

Table 2: Performance results of the NID Classifiers

Technique	AC	PC	RC	F1 score
LSTM Classifier	0.9607	0.4674	1.000	0.4099
DNN Classifier	0.8900	0.8970	0.9000	0.8980

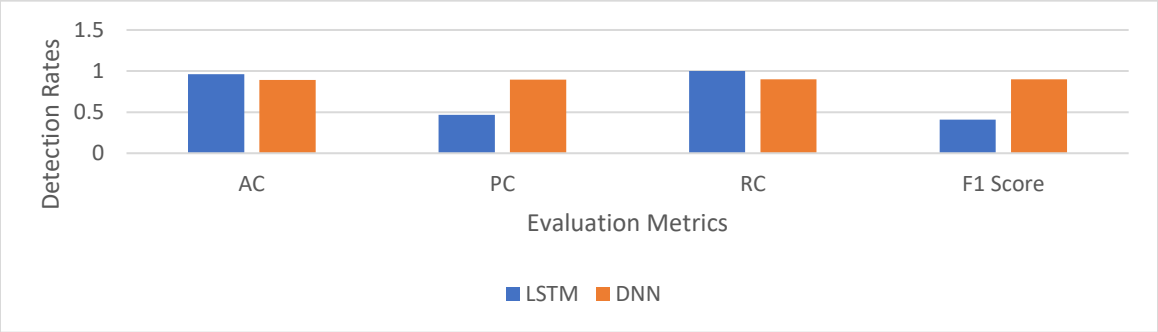

Figure 7: *Test Set Results of the LSTM Classifier and DNN Classifier*

A fitness curve of the proposed LSTM Model has been drawn to capture how the loss and accuracy of the model are taking effect during the 30-epoch run based on training accuracy and validation accuracy. The validation set is solely utilized to assess the performance of the model, while the training set is utilized to train the model. The training loss indicates the rate at which the classifier matches the data for training, while the validation loss implies the rate at which the classifier matches the new data. The fitness curve based on accuracy is displayed in figure 8 and the fitness curve based on Loss is displayed in figure 9.

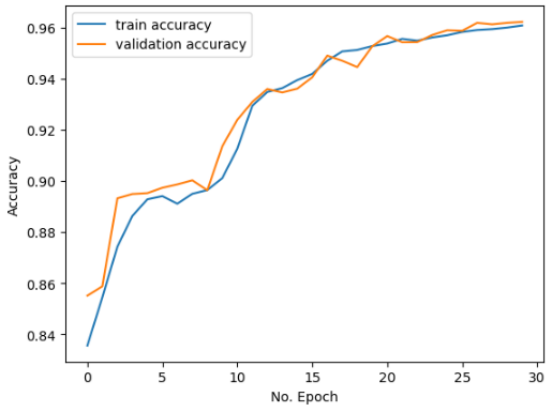

Figure 8*: Fitness curve based on the Accuracy*

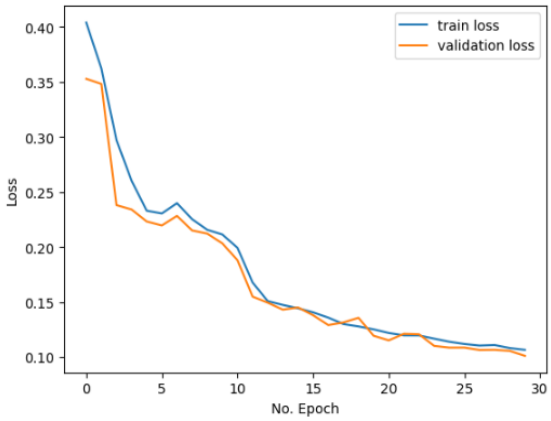

Figure 9: *Fitness curve based on the Loss*

B. Attack based on CTGAN

To generate the adversarial example, the CTGAN synthesizer was used with the help of the ctgan python library. When generating the adversarial data on the dataset, all the attack labels were perturbed since the aim is to make the LSTM model misclassify an attack as a normal record. The success rate of generating the data is recorded based on its accuracy, precision, recall, and F1 score in table 3 compared with the initial results of the target (LSTM) model before the attack. It can be seen that the accuracy, precision, recall, and F1 score of the target model dropped drastically proving the success of the CTGAN attack.

Table 3: Effects of the Attack based on CTGAN

Technique used	AC	PC	RC	F1 score
Target model	0.9554	0.8725	0.3873	0.9210
Target model under attack	0.5257	0.5656	1.0000	0.4099

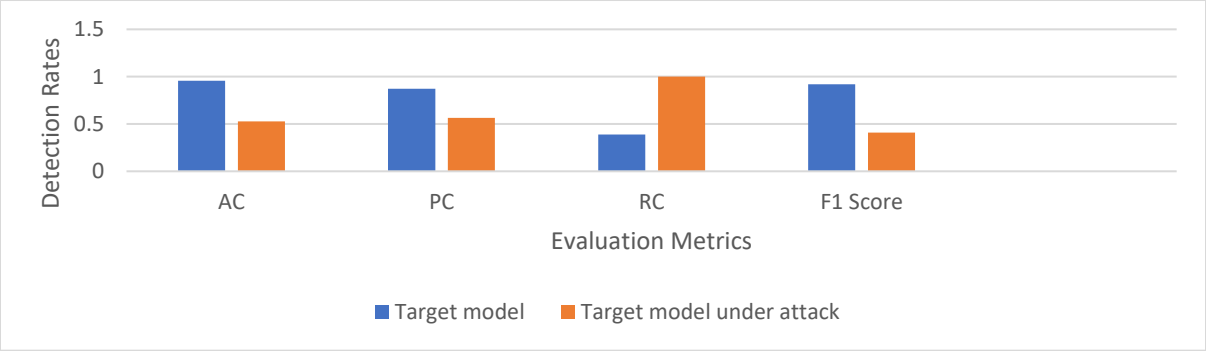

Figure 10: *Impact of Attack Based on CTGAN*

VI. Conclusion

In this study, the viability of generating adversarial samples on deep Long Short-Term Memory-based network intrusion detection systems using the Conditional Tabular Generative Adversarial Network Strategy was investigated. The results of our experiment show how well the LSTM classifier is employed and obtained better performance than that of the DNN Classifier for intrusion detection and how well CTGAN as a black-box attack algorithm performs in terms of Accuracy, Presition, Recall and F1 Score. Compared to the approaches previously addressed in the literature, this research work provided an alternate method of LSTM training with improved accuracy for intrusion detection. These findings suggest possibilities that an adversary could be employed to modify its attacks to trick an intrusion detection classifier.

Acknowledgements

This work has been supported by the Tertiary Education Trust Fund (TETFund) Institutional Based Research (IBR) Fund, through the directorate of Research and Innovation of Nigerian army University, Biu.

References

Aggarwal, C. C. (2018). Neural Networks and Deep Learning. In *Neural Networks and Deep Learning*. https://doi.org/10.1007/978-3-319-94463-0

Alsaeedi, A., & Khan, M. (2019). Performance Analysis of Network Intrusion Detection System using Machine Learning. *International Journal of Advanced Computer Science and Applications*, *10*, 671–678. https://doi.org/10.14569/IJACSA.2019.0101286

Apruzzese, G., & Colajanni, M. (2018). *Evading Botnet Detectors Based on Flows and Random Forest with Adversarial Samples*. https://doi.org/10.1109/NCA.2018.8548327

Apruzzese, G., Colajanni, M., Ferretti, L., & Marchetti, M. (2019). *Addressing Adversarial Attacks Against Security Systems Based on Machine Learning*. https://doi.org/10.23919/CYCON.2019.8756865

Bourou, S., El Saer, A., Velivasaki, T., Voulkidis, A., & Zahariadis, T. (2021). A Review of Tabular Data Synthesis Using GANs on an IDS Dataset. *Information*, *12*, 375. https://doi.org/10.3390/info12090375

Carlini, N., & Wagner, D. (2017). *Towards Evaluating the Robustness of Neural Networks*. https://doi.org/10.1109/SP.2017.49

Chandran, S., & Kumar, K. (2018). A survey of intrusion detection techniques. *International Journal of Engineering & Technology*, *7*, 187. https://doi.org/10.14419/ijet.v7i2.4.13036

Dua, S., & Du, X. (2016). Data Mining and Machine Learning in Cybersecurity. *Data Mining and Machine Learning in Cybersecurity*. https://doi.org/10.1201/b10867

Gao, N., Gao, L., Gao, Q., & Wang, H. (2015). An Intrusion Detection Model Based on Deep Belief Networks. *Proceedings - 2014 2nd International Conference on Advanced Cloud and Big Data, CBD 2014*, 247–252. https://doi.org/10.1109/CBD.2014.41

Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and harnessing adversarial examples. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, *November*.

Heng, L., & Weise, T. (2019). Intrusion Detection System Using Convolutional Neuronal Networks: A Cognitive Computing Approach for Anomaly Detection based on Deep Learning. *Proceedings of 2019 IEEE 18th International Conference on Cognitive Informatics and Cognitive Computing, ICCI*CC 2019*, 34–40. https://doi.org/10.1109/ICCICC46617.2019.9146088

Huang, W., Peng, X., Shi, Z., & Ma, Y. (2020). Adversarial Attack against LSTM-based DDoS Intrusion Detection System. *Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI*, *2020-Novem*, 686–693. https://doi.org/10.1109/ICTAI50040.2020.00110

Jha, S., Tan, K., & Maxion, R. A. (2001). Markov chains, classifiers, and intrusion detection. *Proceedings of the Computer Security Foundations Workshop*, 206–219. https://doi.org/10.1109/CSFW.2001.930147

Kaur, H., & Gill, N. (2013). Host-based Anomaly Detection using Fuzzy Genetic Approach (FGA). *International Journal of Computer Applications*, *74*(20), 5–9. https://doi.org/10.5120/13024-0026

KHORRAM, T. (2021). Network Intrusion Detection using Optimized Machine Learning Algorithms. *European Journal of Science and Technology*, *25*, 463–474. https://doi.org/10.31590/ejosat.849723

Laghrissi, F. E., Douzi, S., Douzi, K., & Hssina, B. (2021). Intrusion detection systems using long short-term memory (LSTM). *Journal of Big Data*, *8*(1). https://doi.org/10.1186/s40537-021-00448-4

Lakshminarayana, D., Philips, J., & Tabrizi, N. (2019). *A Survey of Intrusion Detection Techniques*. https://doi.org/10.1109/ICMLA.2019.00187

Lin, Z., Shi, Y., & Xue, Z. (2018). *IDSGAN: Generative Adversarial Networks for Attack Generation against Intrusion Detection*. 1–16. http://arxiv.org/abs/1809.02077

Martins, N., Cruz, J., Cruz, T., & Henriques Abreu, P. (2020). Adversarial Machine Learning Applied to Intrusion and Malware Scenarios: A Systematic Review. *IEEE Access*, *PP*, 1. https://doi.org/10.1109/ACCESS.2020.2974752

Martins, N., Cruz, J. M., Cruz, T., & Abreu, P. H. (2019). Analyzing the Footprint of Classifiers in Adversarial Denial of Service Contexts. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *11805 LNAI*(July 2021), 256–267. https://doi.org/10.1007/978-3-030-30244-3_22

Moosavi-Dezfooli, S. M., Fawzi, A., & Frossard, P. (2016). DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2016-Decem*, 2574–2582. https://doi.org/10.1109/CVPR.2016.282

Papernot, N., Mcdaniel, P., Jha, S., Fredrikson, M., Celik, Z. B., & Swami, A. (2016). The limitations of deep learning in adversarial settings. *Proceedings - 2016 IEEE European Symposium on Security and Privacy, EURO S and P 2016*, 372–387. https://doi.org/10.1109/EuroSP.2016.36

Ren, K., Zheng, T., Qin, Z., & Liu, X. (2020). Adversarial Attacks and Defenses in Deep Learning. *Engineering*, *6*. https://doi.org/10.1016/j.eng.2019.12.012

Rigaki, M., & Elragal, A. (2017). Adversarial Deep Learning Against Intrusion Detection Classifiers. *CEUR Workshop Proceedings*, *2057*(December), 35–48.

Shende, S. (2020). Long Short-Term Memory (LSTM) Deep Learning Method for Intrusion Detection in Network Security. *International Journal of Engineering Research And*, *V9*. https://doi.org/10.17577/IJERTV9IS061016

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., & Fergus, R. (2014). Intriguing properties of neural networks. *2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings*, *April 2014*.

Wang, Z. (2018). Deep Learning-Based Intrusion Detection with Adversaries. *IEEE Access*, *6*, 38367–38384. https://doi.org/10.1109/ACCESS.2018.2854599

Wu, S. X., & Banzhaf, W. (2010). The use of computational intelligence in intrusion detection systems: A review. *Applied Soft Computing Journal*, *10*(1), 1–35. https://doi.org/10.1016/j.asoc.2009.06.019

Yan, Q., Wang, M., Huang, W., Luo, X., & Yu, F. R. (2019). Automatically synthesizing DoS attack traces using generative adversarial networks. *International Journal of Machine Learning and Cybernetics*, *10*(12), 3387–3396. https://doi.org/10.1007/s13042-019-00925-6

Yang, K., Liu, J., Zhang, C., & Fang, Y. (2019). Adversarial Examples Against the Deep Learning Based Network Intrusion Detection Systems. *Proceedings - IEEE Military Communications Conference MILCOM*, *2019-Octob*, 559–564. https://doi.org/10.1109/MILCOM.2018.8599759

Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. *IEEE Access*, *5*, 21954–21961. https://doi.org/10.1109/ACCESS.2017.2762418