Volume 3, Issue X, October 2024, No. 48, pp. 644-654
Submitted 8/10/2024; Final peer review 1/11/2024
Online Publication 4/11/2024
Available Online at http://www.ijortacs.

A SMART ADVERSARIAL THREATS DETECTION SYSTEM USING HYBRID OF BIGAN-BIGRU TECHNIQUE

[1*]Ojiako F.N., [2]Mgbeafulike I.J., [3]Okeke O.C
[1,2,3]Department of Computer Science, Faculty of Physical Sciences,
Chukwuemeka Odumegwu Ojukwu University, Anambra State
Author Email: [1]ojiakofranklin4@gmail.com, [2]Ike.mgbeafulike@gmail.com
[3]cj.okonkwo@coou.edu.ng.com

Abstract

The study presents modelling a smart adversarial threats detection system using hybrid of Bi-Directional Generative Adversarial Network (Bi-GAN) and Bi-Directional Gated Recurrent Unit (Bi-GRU). The aim is to integrate Bi-GAN model in the network to capture complex and dynamic threat patterns. The primary objective is to develop a data model that effectively processes network threat features and generates a high-performing detection rate for adversarial threats. The methodology used is dynamic system development method. The collected threat data were pre-processed using analysis of variance technique for feature selection and then Convolutional Neural Network (CNN) for feature transformation. The features were then used to train the Bi-GAN, which was designed to learn temporal and sequential characteristics of adversarial attack patterns, making it ideal for real-time network monitoring. The model was implemented using python programming language and simulated in Kali-linux and Virtual lab, environment. The result of the study suggests that the Bi-GAN model is highly effective at detecting adversarial attacks. The results demonstrated that the Bi-GAN model considered adversarial attack vector such as denial of service, ip-sweep, flood, sql-injection attack and legitimate packets. The experimental findings showed that the proposed system could reliably identify all the attack vectors IP addresses and classifying them as threat, while allowing throughput for only the legitimate packets. In conclusion, this research contributes to the field of cybersecurity by providing a novel framework for addressing the limitations of existing intrusion detection systems.

Keywords: Adversarial Threats; Intrusion Detection System; Wireless Network; Bi-Directional Generative Adversarial Network; Convolutional Neural Network

1. INTRODUCTION

Over the years, cyber security has become increasingly important for the security of data and network infrastructures. On daily basis, more and more people are interconnected through the

internet, exchanging crucial information which in most cases needs maximum protection both online and offline, thus presenting the need for cybersecurity (Kamalakannan et al., 2023).

According to Maithem and Al-Sultany (2021), cybersecurity is the process of safeguarding this digital information from unauthorized access. This can be done through network security, application security, data security, endpoint security, cloud security, management of vulnerability, compliance and regulations, access control management and incident response systems (Diamantopoulou, et al., 2020). To ensure this security compliance, the National Institute of Standards and Technology (NIST SP 800-160) and the International Standards and Organization (ISO-73:2009), have provided standardized frameworks such as ISO 27005, ISO 27001, ISO 27006, and NIST SP-800-37, which allow organizations to comply with safety requirements and mitigate risk (Cheimonidis et al., 2023), however the changing landscapes of threats, emerging vulnerabilities, adversarial nature of attack have made this frameworks not reliable to provide maximum security as required (Frank et al., 2019).

According to Cheimonidis et al., (2023), the static nature of existing cyber security frameworks are a major weakness exploited by threat actors to perpetrate cyber-attack, and this has remained a major challenge. Cyber-attack is an offensive action by a hacker, insider threat, or unauthorized individual to gain access to network facility housing important information and comprise the integrity (Kurochkin and Volkov, 2020). To perform this attack, first the vulnerability in the network are identified, then exploited through advance programming, before penetrating through threat vectors like malware attack, phishing attack, social engineering attack, zero-day attack, etc, hence presenting the need or Intrusion Detection System (IDS) (Mohammad et al., 2024).

IDS are specialized tools designed to detect unauthorized access or malicious features trying to penetrate a network infrastructure (Wei and Shangguan, 2023). The IDS are of two main types which are the host based and network based. The host based is tailored towards the protection of network infrastructure from the end points, while the network-based monitors the network to detect and isolate malicious threat vectors (Vinayakumar et al., 2019).

Overall, both the signature based and anomaly-based approach of IDS need room for improvement despite their success, and hence presents the need for this research an intrusion detection system using deep learning technique (Ahmad et al., 2020). Deep learning is a family of neural network with many hidden layers, and due to their effectiveness in classification and prediction problems, has been applied to cybersecurity (Wei and Shangguan, 2023). To achieve this presents a Bi-Directional Generative Adversarial (BI-GAN) model is used for intrusion detection system in a network. The BI-GAN will be applied for data augmentation to formulate a new data model with adversarial characteristics to capture uncertain threat features. This method will be employed in this study to present an improved IDS capable of safeguarding network infrastructures in real-time.

2. RESEARCH METHODOLOGY

The methodology used for the study is the Dynamic System Development Method (DSDM). This approach was chosen due to its iterative and incremental nature, which allows for flexibility and adaptability in response to changing project requirements. DSDM emphasizes user

involvement and collaboration, ensuring that the developed system meets the actual needs of stakeholders. It also promotes frequent delivery of functional components, enabling continuous feedback and improvements. This methodology is particularly effective in managing risks and ensuring the project stays on track, making it an ideal choice for the dynamic environment of the study. To achieve this methodology, a new data model of IDS was created and then applied after processing to train a deep learning algorithm and generate a model for IDS.

2.1 Data Description

The database tool used is MySQL dataset and the structured table presenting the attributes, their data formats, and a description based of the dataset is presented in Table 1. This table helps describe each attribute, the data format (either integer, float, categorical, or binary), and a brief explanation of its meaning in the context of network traffic and intrusion detection systems.

Table 1: Data description table

Attribute	Data Format	Description
duration	Integer	Length (in seconds) of the connection
protocol_type	Categorical	Type of protocol used (e.g., TCP, UDP, ICMP)
service	Categorical	Network service on the destination (e.g., HTTP, FTP, SSH)
flag	Categorical	Status of the connection (e.g., SF, REJ)
src_bytes	Integer	Number of data bytes sent from source to destination
dst_bytes	Integer	Number of data bytes sent from destination to source
land	Binary (0 or 1)	1 if the connection is to/from the same host/port, 0 otherwise
wrong_fragment	Integer	Number of wrong fragments in this connection
urgent	Integer	Number of urgent packets
hot	Integer	Number of hot indicators (features representing suspicious activity)
num_failed_logins	Integer	Number of failed login attempts
logged_in	Binary (0 or 1)	1 if successfully logged in, 0 otherwise
num_compromised	Integer	Number of compromised conditions
root_shell	Binary (0 or 1)	1 if root shell is obtained, 0 otherwise
su_attempted	Binary (0 or 1)	1 if "su root" command attempted, 0 otherwise
num_root	Integer	Number of root accesses or privileged file accesses
num_file_creations	Integer	Number of file creation operations
num_shells	Integer	Number of shell prompts invoked
num_access_files	Integer	Number of operations on access control files
num_outbound_cmds	Integer	Number of outbound commands in an FTP session (always 0 in this dataset)
is_host_login	Binary (0 or 1)	1 if the login is to the host, 0 otherwise
is_guest_login	Binary (0 or 1)	1 if the login is a guest login, 0 otherwise
count	Integer	Number of connections to the same host as the current connection in the past 2 seconds

srv_count	Integer	Number of connections to the same service as the current connection in the past 2 seconds
serror_rate	Float	% of connections that have "SYN" errors
srv_serror_rate	Float	% of connections to the same service that have "SYN" errors
rerror_rate	Float	% of connections that have "REJ" errors
srv_rerror_rate	Float	% of connections to the same service that have "REJ" errors
same_srv_rate	Float	% of connections to the same service
diff_srv_rate	Float	% of connections to different services
srv_diff_host_rate	Float	% of connections to different hosts
dst_host_count	Integer	Number of connections to the same destination host in the past 100 connections
dst_host_srv_count	Integer	Number of connections to the same service as the current connection in the past 100 connections
dst_host_same_srv_rate	Float	% of connections to the same service among the connections to the destination host
dst_host_diff_srv_rate	Float	% of connections to different services among the connections to the destination host
dst_host_same_src_port_rate	Float	% of connections to the same source port among the connections to the destination host
dst_host_srv_diff_host_rate	Float	% of connections to different destination hosts among the connections to the same service
dst_host_serror_rate	Float	% of connections to the destination host that have "SYN" errors
dst_host_srv_serror_rate	Float	% of connections to the same service as the current connection that have "SYN" errors
dst_host_rerror_rate	Float	% of connections to the destination host that have "REJ" errors
dst_host_srv_rerror_rate	Float	% of connections to the same service as the current connection that have "REJ" errors
attack	Categorical	Type of attack (or "normal" for no attack)
level	Integer	Difficulty level for detecting the attack (specific to the dataset and experiments)

Table 1 presents the data structure which was then augmented to generate adversarial sample using Bi-Directional Generative Adversarial Network (Bi-GAN). Bi-GAN is proposed as an adversarial attack feature generator in cybersecurity systems to enhance detection models. By leveraging its encoder and generator, Bi-GAN can produce synthetic adversarial attack features that closely resemble real-world attack patterns, including DDoS, SQL injection, and Trojan attacks. The discriminator works to distinguish between these synthetic features and actual attack data, pushing the generator to create highly realistic adversarial examples. This approach provides a robust method to simulate diverse attack scenarios, improving the resilience of detection systems by training them on a wider range of attack behaviours and features.

2.2 Feature Extraction using CNN

CNN was used to extract important and meaningful features from input data of adversarial attack by passing it through several layers of convolution, pooling, and activation functions as shown in the CNN block diagram in Figure 1, while the architecture was presented in Table 2.

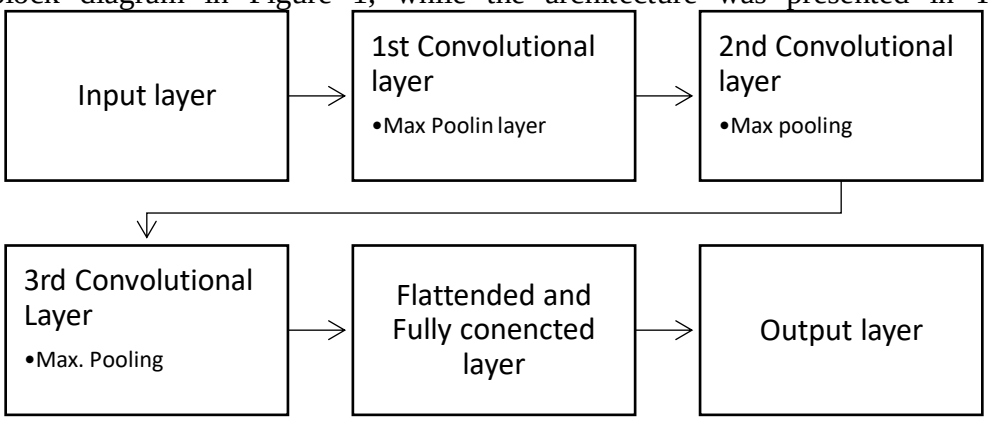

Figure 1: Block diagram of the CNN

Table 2: Architecture of CNN

Layer Type	Description	Output Shape	Number of Parameters
Input Layer	Input image data (1x224x3)	(224, 224, 3)	0
Conv2D	32 filters, 3x3 kernel, ReLU activation	(224, 224, 32)	896
MaxPooling2D	2x2 pool size	(112, 112, 32)	0
Conv2D	64 filters, 3x3 kernel, ReLU activation	(112, 112, 64)	18,496
MaxPooling2D	2x2 pool size	(56, 56, 64)	0
Conv2D	128 filters, 3x3 kernel, ReLU activation	(56, 56, 128)	73,856
MaxPooling2D	2x2 pool size	(28, 28, 128)	0
Flatten	Flattens the 3D tensor into 1D vector	(100352)	0
Fully Connected	512 units, ReLU activation	(512)	51,180,544
Output Layer	Output feature vector for downstream tasks	(512)	0

Table 1 of CNN feature extractor represents a structured breakdown of the layers involved in the CNN architecture as shown in Figure 1. It starts with an Input Layer which dimension the input threat features into 1 Dimension (1D). This is followed by multiple Convolutional Layers (Conv2D) with filters and kernels that detect spatial features, such as edges and patterns, and MaxPooling Layers that downsample the feature maps, reducing dimensionality while preserving important information. After several layers of convolution and pooling, the Flatten Layer converts the 3D feature maps into a 1D vector. The Fully Connected (Dense) Layer then processes these features, outputting a feature vector, which can be used for downstream tasks like classification. The number of parameters in each layer shows the complexity of the learned features. This architecture efficiently extracts hierarchical features from the input data.

Corresponding Author Tel: + 23408035971621

3 THE PROPOSED BI-GAN BASED IDS SYSTEM

The proposed system will be developed using through the following steps. First a data model which captures the dynamic characteristics of threat features within a network will be collected. Then to capture the adversarial samples of the data, the Bi-GAN technique will be applied which will be integrated to form the comprehensive IDS dataset. To address issues of data balancing and quality, data processing steps such as imputation and SMOTE will be applied, while feature selection ensure prioritization of important features and then transformed into a sequential data for training Bi-GAN and generation of model for IDS. The proposed system block diagram was presented in Figure 2.

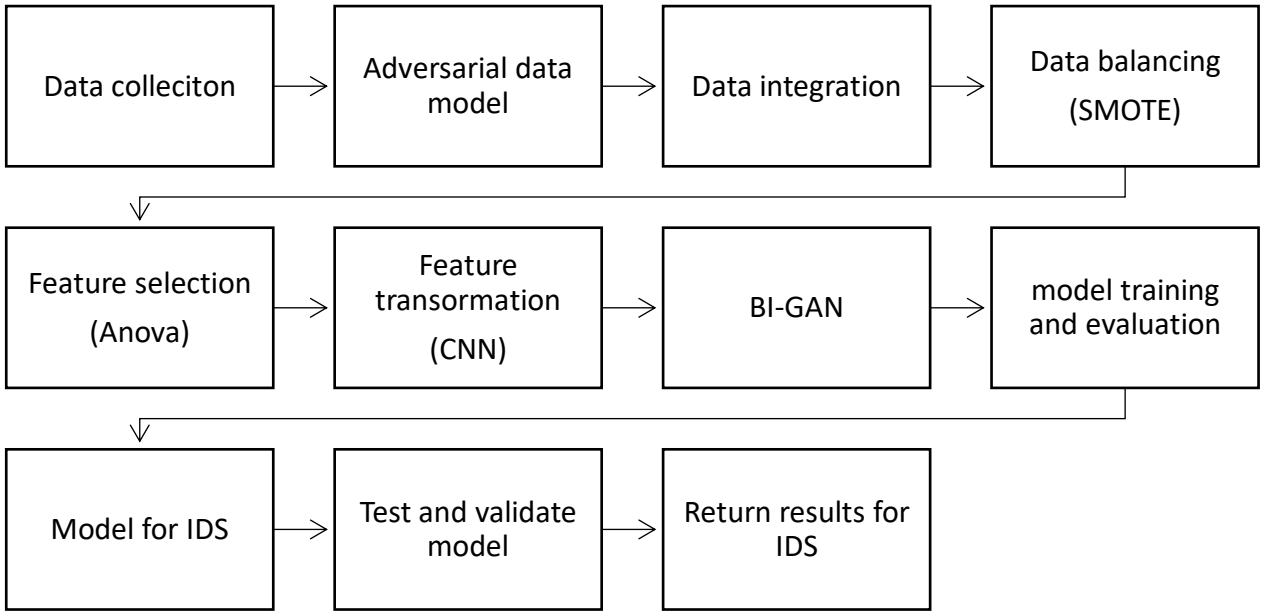

Figure 2: Block diagram of the proposed IDS with Bi-GAN

The Figure 2 presents the block diagram of the IDS using Bi-GAN. Initially data collected was perturbated with Bi-GAN to generate a new data model which captures the adversarial signature of the threat. To integrate the dataset as a new data model, the two threat features will be combined in excel environment and then generate the database for IDS considering emerging threats and adversarial attack. To apply processing on the data and manage data imbalance, SMOTE was used, while ANOVA was applied to identify and select the most important features on the dataset, before applying CNN for data transformation. The transformed data will be applied to train a Bi-GAN architecture and generate IDS model. During the training of the model, it will be evaluated through testing and validation before deployment for IDS.

3.1 Algorithms of the IDS Model

This section presents the stepwise algorithms for ANOVA feature selection, CNN and Bi-GAN respectively.

ANOVA Feature Selection Algorithm

1) *Input: Dataset X with features* $(x_1, x_2, \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots x_n)$
2) *Target variable* (Y)

3) *Initialization: Prepare a list to store F-statistics for each feature.*
4) *For each feature* x_i
5) *Group the data by the unique values of Y.*
6) *Calculate the mean for each group:* $Mean_j = \frac{1}{N_j} \sum_{k=1}^{N_j} x_{ij}$ *: where* N_j *is number of sample in group j.*
7) *Calculate the overall mean of the feature:* $O_m = \frac{1}{N_j} \sum_{j=1}^{m} N_j . mean_j$
8) *Calculate the between-group variance (SSB):* $SSB = \sum_{j=1}^{m} N_j \left(mean_j - O_m\right)^2$*:*
9) *Calculate the within-group variance (SSW):* $SSW = \sum_{j=1}^{m} \sum_{k=1}^{N_j} \left(x_{ik} - mean_j\right)^2$
10) *Calculate the F-statistic:* $F = \frac{\frac{SSB}{m-1}}{\frac{SSW}{N-m}}$
11) *Store the F-statistic for feature* x_i
12) *Rank all features based on their F-statistic values in descending order.*
13) *Select the top k feature based on a predefined threshold.*
14) *End*

Convolutional Neural Network (CNN) Algorithm

1. *Input: Image data Xwith dimensions* $H * W * C$
2. *1: Initialize the architecture*
3. *Define the number of convolutional layers, pooling layers, and fully connected layers.*
4. *For each convolutional layer*
5. *2: Apply convolution with K filters of size* $F * F$ *to the input image*
6. *Perform convolution operation* $Y_{i,j} = \sum_{m=1}^{F} \sum_{n=1}^{F} X_{i+m,j+n} . K_{m,n}$
7. *Apply RELU activation function* $\sigma\left(Y_{i,j}\right) = max(0, Y_{i,j})$
8. *Apply pooling layer (max pooling).*
9. *Flatten the output from the last pooling layer.*
10. *Pass through fully connected layers.*
11. *Output: Class probabilities via SoftMax activation.*
12. *Define loss function (cross-entropy).*
13. *Back-propagation:*
14. *Compute gradients and update weights using an optimization algorithm (e.g., Adam)*
15. *Bidirectional Generative Adversarial Network (Bi-GAN) Algorithm*
16. *End*

Stepwise Algorithm: Bidirectional Generative Adversarial Network (Bi-GAN)

1) *Input: Real data samples Xand noise vector Z*
2) *Initialize Generator B Discriminator Dand Encoder E, Training Loop (for each iteration):*
3) *Sample real data X and noise Z*
4) *Train Discriminator D*
5) *Maximize:* $L_D = \mathbb{E}x \sim pdata\ [logD(X)] + \mathbb{E}z \sim pz\ [log\left(1 - D\left(G(Z)\right)\right)]$
6) *Train Generator* G:
7) *Minimize* $L_G = \mathbb{E}z \sim pz\ [log\left(1 - D\left(G(Z)\right)\right)]$

8) Train Encoder E
9) Minimize $L_E = \mathbb{E}x{\sim}pdata\,[logD(E(X))]$
10) Repeat steps 3 to 5 until convergence.
11) Output: Generate samples $G(Z)$*that resemble real data* X*.*
12) End

4 SYSTEM IMPLEMENTATION

The system implementation of adversarial attack detection using a Bi-GAN model can be achieved by integrating Python with the Kali-Linux operating system and a virtual lab for testing denial of service (DoS) attacks. The Bi-GAN model can be developed in Python using deep learning libraries like TensorFlow or PyTorch. After training the model on adversarial attack data, it can be deployed in Kali-Linux, a security-focused Linux distribution. In the virtual lab environment, network traffic can be simulated to create realistic DoS attack scenarios. This lab setup allows the system to monitor incoming traffic, apply the Bi-GAN model for attack detection, and evaluate its performance against real-time adversarial attacks. The integration of Python with Kali-Linux enables automated testing and real-time attack detection, allowing the system to assess the model's robustness and responsiveness to adversarial inputs in a controlled environment.

5 PERFORMANCE EVALUATION OF THE IDS MODEL

The performance evaluation of the adversarial attack model was presented in this section considering different adversarial attack vectors such as DDoS, IP_sweep, flood attack and legitimate packets. The figure 3 presents the data distribution.

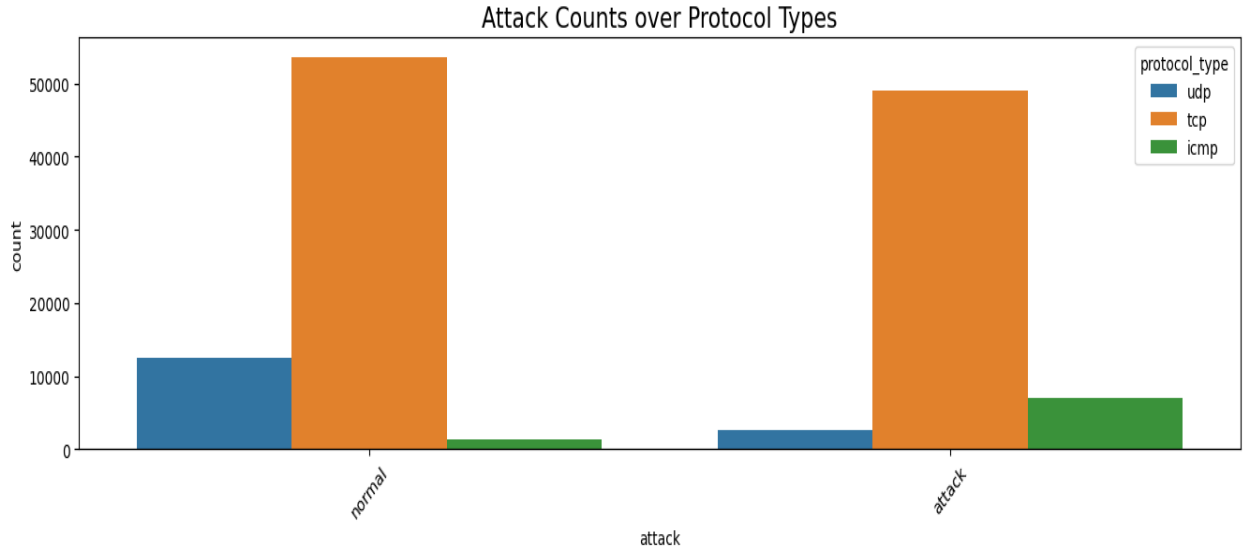

Figure 3: distribution of data classes across different protocols

The figure 3 presents the class distribution of packet data across different protocols and classes. This data was distributed according to features for normal packet and threat vectors as shown in the figure 4, before the most important features were prioritized using the ANOVA approach in figure 5.

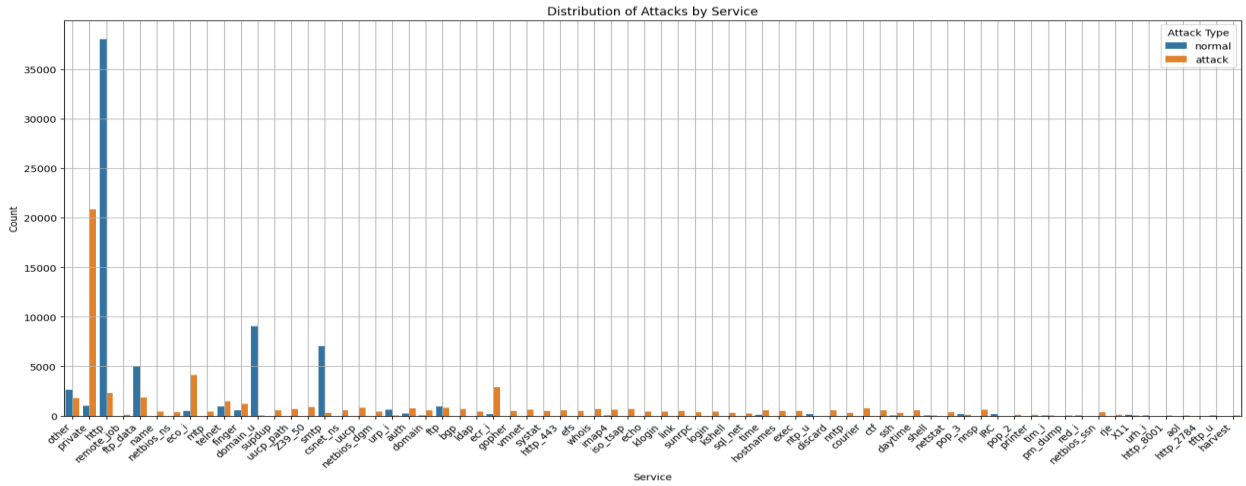

Figure 4: Distribution of packet features for attack and normal user

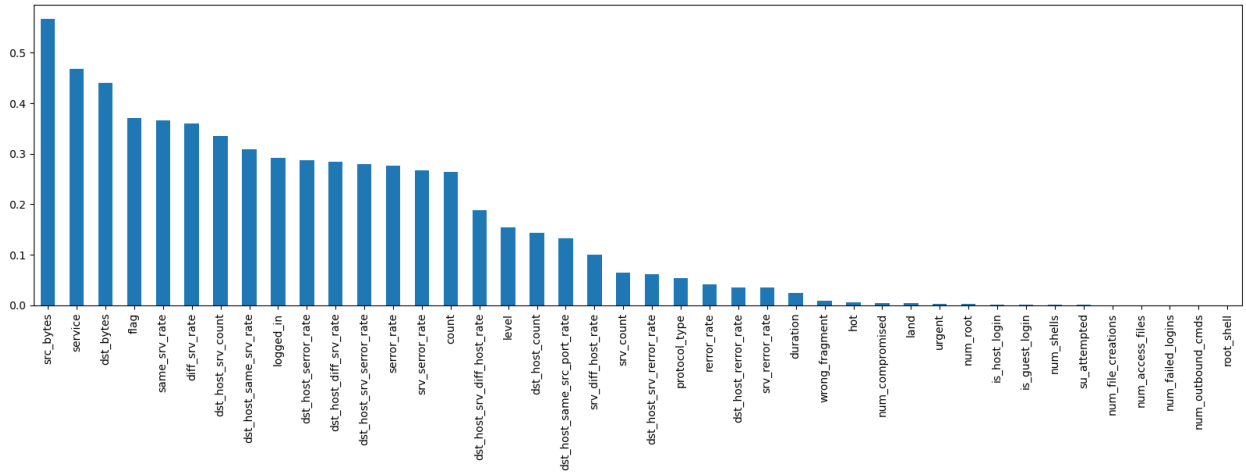

Figure 5: Result of the feature identification by ANOVA

The figure 5 reported the performance of the ANOVA for feature selection. This approach computes the fitness of each attribute and then score and presents according to the rankings in descending order, with the most important at the top priority. The data were then converted to 1D using the CNN as a compact feature vectors and feed to train the Bi-GRU model. During the training, the performance of the Bi-GRU model was evaluated considering accuracy, loss, precision and recall. The Figure 6 presents the result of the adversarial attack detection model using flood feature vectors from 41 different attackers. From the results, it was observed that all the 41 IP address of attackers were correctly detected by the hybrid deep learning based adversarial attack detection model. This suggested that our model have high detection rate for flood attack. In another experiment, 25 IP_sweep packets were applied to test the hybrid deep learning based adversarial attack detection model as shown in the Figure 7.

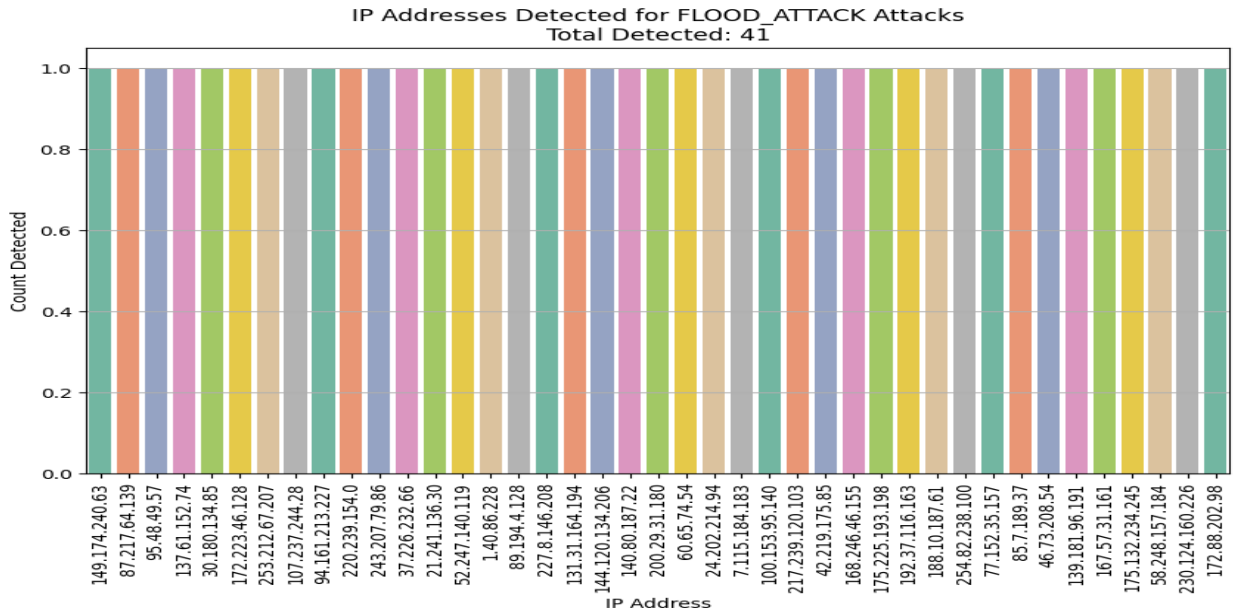

Figure 6: Result of the adversarial attack detection model considering flood threat vectors

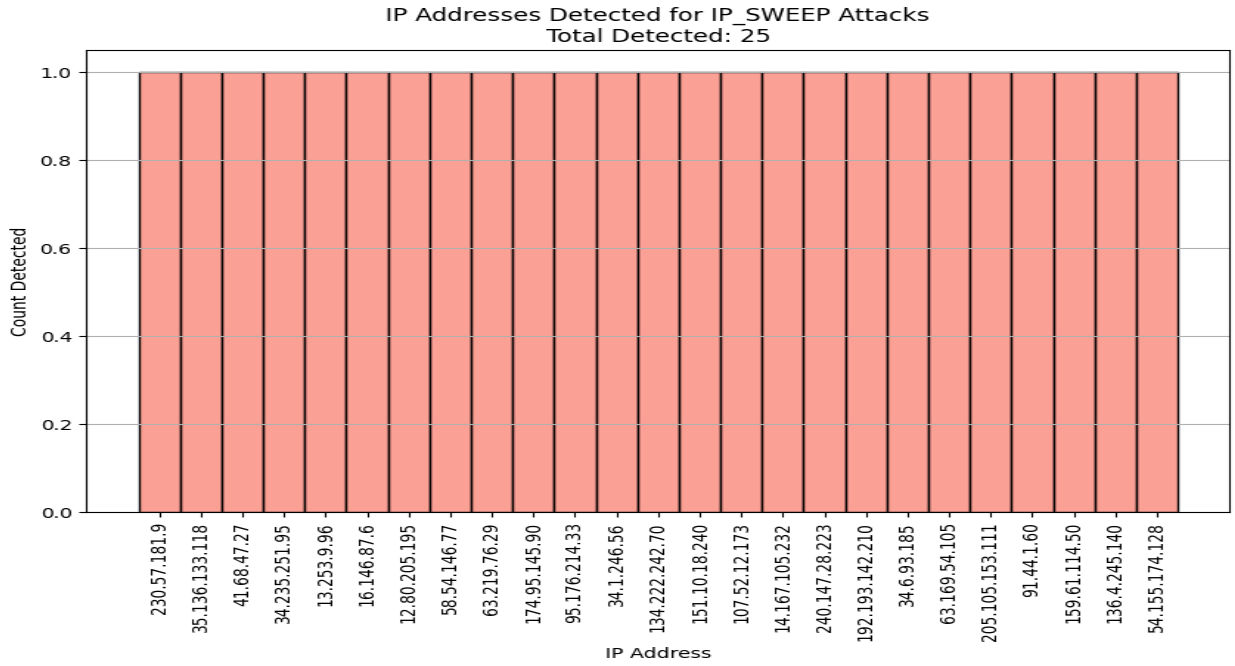

Figure 7: Result of experiment with IP-sweep

Figure 7 presents the results of the IP sweep experiment on the hybrid deep learning based adversarial attack detection model. During the attack, it was observed that the 25 threat IP addresses were correctly classified as intrusion by our model. Figure 8 reported the hybrid deep learning based adversarial attack detection model when evaluated against adversarial DDoS attack.

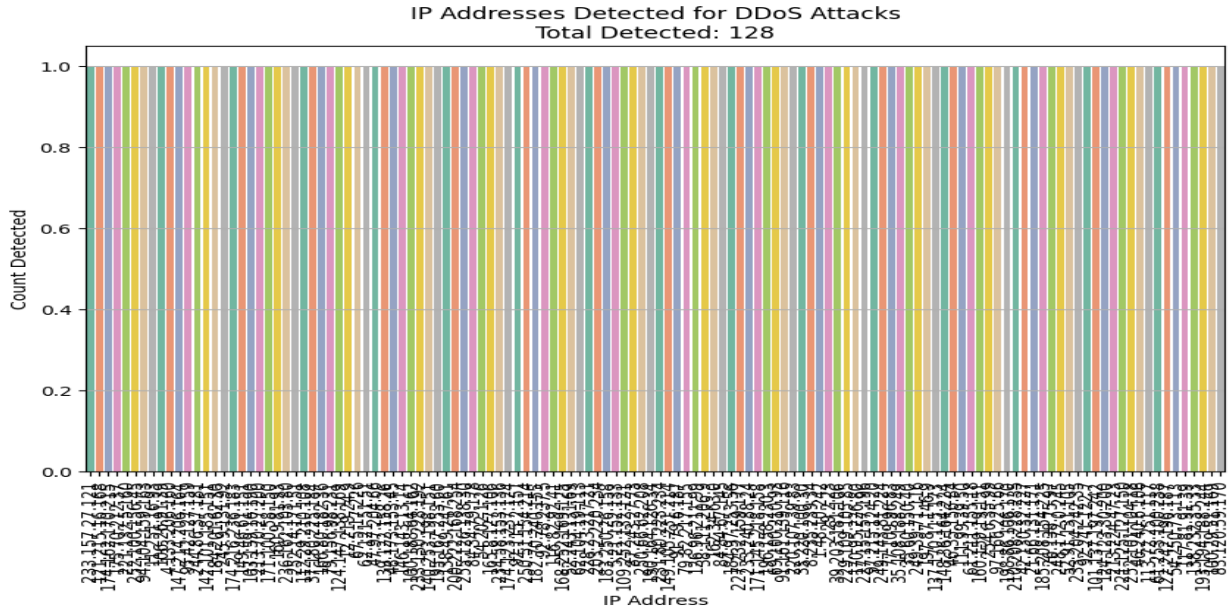

Figure 8: Results of our model against adversarial DDOS attack

The Figure 8 reported the performance of the hybrid deep learning based adversarial attack detection model when evaluated against adversarial DDoS attack. From the results, it was observed that our model was able to correctly classify the IP of the DDoS agents and isolate from the network. In the Figure 9, the performance of our model against SQL injection attack was evaluated and reported for analysis. This SQL injection attack was aimed at obstructing services in the database of network server. The impact of the hybrid deep learning based adversarial attack detection model on this attack was reported.

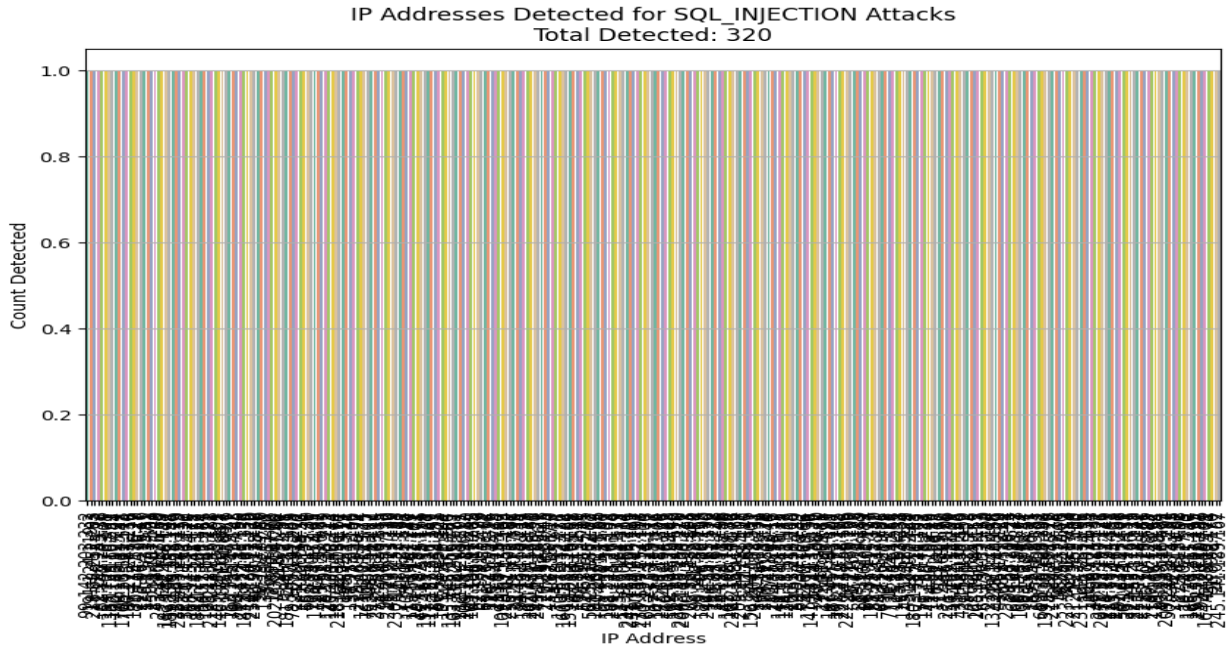

Figure 9: Result of our model against SQL injection attack

Figure 9 presents the performance of the hybrid deep learning based adversarial attack detection model considering SQL injection attack. The result showed that our model was able to correctly classify all the IP addresses used for the attack. Also, the results when evaluated with legitimate packets were also evaluated with our model and it was observed that their IP were correctly classified as normal packets. Overall, it can be concluded that our model is reliable to detect different adversarial attacks and mitigate threat impact of the network infrastructure.

6 CONCLUSION

The research successfully demonstrated the potential of using deep learning models, specifically Bi-GAN, to detect network intrusions in real-time. By integrating adversarial learning techniques, such as Bi-GAN, the system was able to simulate a wide array of attack strategies, making the detection model more resilient against evolving threats. The use of data processing techniques played a vital role in ensuring that the model was not overwhelmed by irrelevant data, allowing it to focus on key features that defined various types of intrusions. The paper highlighted the importance of integrating both generative and discriminative models in cybersecurity applications. While Bi-GAN enabled the system to understand how attacks could evolve it also allowed for the detection of these attacks in real-time by recognizing recurring patterns within the data. Despite the achievements of the technique, the study also acknowledged limitations, such as the system's reliance on large datasets and the complexity of model training.

7. REFERENCES

Ahmad Z., Khan A., Shiang C., Abdullah J., & Ahmad F., (2020) Network intrusion detection system: A systematic study of machine learning and deep learning approaches. Trans Emerging Tel Tech. 2021;32:e4150. wileyonlinelibrary.com/journal/ett 1 of 29 https://doi.org/10.1002/ett.4150

Cheimonidis P., &Rantos K., (2023) Dynamic Risk Assessment in Cybersecurity: A Systematic Literature Review. *Future Internet* **2023**, *15*, 324. https://doi.org/10.3390/fi15100324

Diamantopoulou V., Aggeliki T., & Maria K., (2020) From ISO/IEC27001: 2013 and ISO/IEC27002: 2013 to GDPR compliance controls. Information & Computer Security 28: 645–62.

Frank M.L., Jonathan H.G., & Jonathan S.P., (2019) How disclosing a prior cyberattack influences the efficacy of cybersecurity risk management reporting and independent assurance. Journal of Information Systems 33: 183–200

Kamalakannan D., MenagaD., ShobanaS., Daya-SagarK., Rajagopal R.,&TiwariM., (2023) A Detection of Intrusions Based on Deep Learning, Cybernetics and Systems, DOI: 10.1080/01969722.2023.2175134

Kurochkin I., & Volkov S., (2020) Using GRU based deep neural network for intrusion detection in software-defined networks: IOP Conference Series: Materials Science and Engineering. 927. 012035. 10.1088/1757-899X/927/1/012035.

Maithem M., & Al-Sultany G., (2021) Network intrusion detection system using deep neural networks. ICMAICT 2020 Journal of Physics: Conference Series 1804 (2021) 012138 IOP Publishing doi:10.1088/1742-6596/1804/1/012138

Mohammad R., Saeed F.,Almazroi A.,Alsubaei F., &Almazroi A., (2024) Enhancing Intrusion Detection Systems Using a Deep Learning and Data Augmentation Approach. Systems 2024, 12, 79. https://doi.org/10.3390/systems12030079

Vinayakumar R., Mamoun A., Soman K., Prabaharan P., Ameer A., &Sitalakshmi V., (2019) Deep Learning Approach for Intelligent Intrusion Detection System. DOI 10.1109/ACCESS.2019.2895334, IEEE Access

Wei Y., & Shangguan M., (2023) A review of deep learning based intrusion detection systems. Highlights in Science, Engineering and Technology AICT 2023 Volume 56 (2023)