G-GANS for Adaptive Learning in Dynamic Network Slices

Meshari Huwaytim Alanazi
Department of Computer Science, Northern Border University, Saudi Arabia
meshari.alanazi@nbu.edu.sa (corresponding author)

Received: 11 February 2024 | Revised: 24 March 2024 | Accepted: 8 April 2024

 DOI: https://doi.org/10.48084/etasr.7046

ABSTRACT

This paper introduces a novel approach to improve security in dynamic network slices for 5G networks using Graph-based Generative Adversarial Networks (G-GAN). Given the rapidly evolving and adaptable nature of 5G network slices, traditional security mechanisms often fall short in providing real-time, efficient, and scalable defense mechanisms. To address this gap, this study proposes the use of G-GAN, which combines the strengths of Generative Adversarial Networks (GANs) and Graph Neural Networks (GNNs) for adaptive learning and anomaly detection in dynamic network environments. The proposed approach utilizes GAN to generate realistic network traffic patterns, both normal and adversarial, whereas GNNs analyze these patterns within the context of the network's graph-based topology. This combination facilitates the early detection of anomalies and potential security threats, adapting to the ever-changing configurations of network slices. The current study presents a comprehensive methodology for implementing G-GAN, including system architecture, data processing, and model training. The experimental analysis demonstrates the efficacy of G-GAN in accurately identifying security threats and adapting to new scenarios, revealing that G-GAN outperformed established models with an accuracy of 97.12%, precision of 96.20%, recall of 97.24%, and F1-Score of 96.72%. This study not only contributes to the field of network security in the context of 5G, but also opens avenues for future exploration in the application of hybrid AI models for real-time security across various domains.

Keywords-accuracy; adaptively learning; anomaly detection; generative adversarial networks; graph neural networks

I. INTRODUCTION

The rapid evolution of wireless communication systems has put forth a demand for a diverse range of services to meet the varying requirements of applications, such as enhanced Mobile Broadband (eMBB), ultra-Peliable and Low-Latency Communication (uRLLC), and massive Machine Type Communication (mMTC). For example, eMBB applications demand high throughput for activities, like virtual reality and video streaming, whereas uRLLC services require low latency and high reliability for critical tasks such as autonomous driving. Similarly, mMTC services, which cater to sensing and monitoring applications, require a widespread connectivity. However, traditional network architectures struggle to effectively meet these diverse service needs. Network slicing is a concept introduced in 5G networks that enables the provision of tailored services with diverse requirements through a unified network infrastructure. Figure 1 shows how the network slicing framework can perform several services concurrently with access, transport, and core network slices. The core network slice includes the control plane and the user plane, offering a mix of shared and exclusive functions across different slices. Functions, such as the Session Management Function (SMF), Mobility Management Function (MMF), User Plane Function (UPF), and Policy Control Function (PCF) are examples of this provision. In particular, key industry players, such as Ericsson and Nokia, have developed and implemented their own versions of network-slicing solutions tailored to their specific market needs [1].

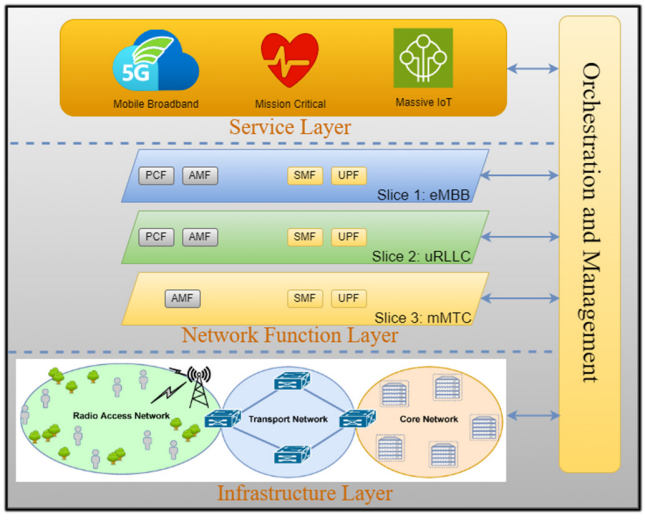

Fig. 1. Network slicing for 5G networks.

The network slicing business model, as illustrated in Table I, includes various essential components and crucial entities for its operation. These entail the Network Slicing Instance (NSI), Network Slicing Subnet Instance (NSSI), logical network, Network Sub-Slice (NSS), Network Slicing Template (NST), network segment, Network Function Virtualization (NFV), Software-Defined Networking (SDN), network slicing manager, communication service manager, resource slice, network slicing provider, network slicing terminal, network slicing tenant, network slicing repository, slice border control, slice selection function, infrastructure owner, infrastructure slice, infrastructure slice provider, and infrastructure slice tenant [2, 3]. Each element plays a specific role in managing the lifecycle of a network slice. For example, the NSI represents a collection of comprehensive logical networks that offer varied services tailored to specific needs, including multiple subslice instances.

TABLE I. KEY COMPONENTS AND ROLES IN NETWORK SLICING ARCHITECTURE

Component	Role
Network Slicing Instance (NSI)	Represents a set of logical networks.
Network Slicing Subnet Instance (NSSI)	Denotes localized logical networks within a slice, shareable across multiple NSIs.
Network Sub-slice (NSS)	A subdivision within an NSI.
Network Slicing Template (NST)	Preset configurations for creating NSIs with predefined characteristics.
Network Function Virtualization (NFV)	Technology enabling the virtualization of network functions for flexible deployment and management.
Software-Defined Networking (SDN)	A networking approach that enables programmable network configurations, aiding in the creation and management of network slices.
Communication Service Management Function (CSMF)	Manages communication and updates slice requirements.
Network Slice Management Function (NSMF)	Manages NSIs and responds to requirements from the CSMF.
Network Slice Subnet Management Function (NSSMF)	Manages NSSIs as specified by the NSMF.

In contrast, NSSI denotes the localized logical network within a slice that can be shared across multiple NSIs. Logical networks are virtual representations of network functions established on a single physical network. The network slicing manager is instrumental in managing the complete lifecycle of each slice or sub-slice through various management functions. These include the Communication Service Management Function (CSMF), Network Slice Management Function (NSMF), and Network Slice Subnet Management Function (NSSMF). CSMFs oversee the management, communication, and the updating of slice requirements to support service requests via the communication service manager. NSMFs manage NSIs based on notifications from CSMFs, whereas NSSMFs handle NSSIs according to NSMF specifications. Network slicing requirements cover various aspects, including network type, capacity, Quality of Service (QoS), latency, security level, device count, and throughput. The resource slice comprises the physical and virtual resources needed for the network slices to function. The network slicing provider owns the physical infrastructure where multiple slices are created, while the network slicing tenant refers to the NSI users who deliver specific services to customers. The infrastructure owner possesses the physical infrastructure, whereas the infrastructure slice provider owns the infrastructure and leases it for hosting a variety of services through network slicing. Infrastructure slice tenants are users of the infrastructure slice itself [4].

A. Background

The emergence of 5G technology signifies a crucial transformation in the telecommunication landscape, introducing speeds, latency, and connectivity of an unprecedented nature. This technology is not merely an incremental advancement in bandwidth but represents a fundamental shift in the fabric of network architecture, opening avenues for groundbreaking applications, namely the Internet of Things (IoT), autonomous vehicles, smart urban ecosystems, and enhanced mobile broadband [5]. At the core of the 5G's operational capability lies the innovative concept of dynamic network slicing. This avant-garde architectural design allows the partitioning of a physical network into multiple, distinct virtual networks. Each of these networks is designed to cater to specific application demands, service requirements, or device utilities, thus ensuring a versatile and efficient network ecosystem [6]. Such an arrangement not only optimizes the network for thousands of service demands, ranging from high bandwidth to paramount reliability, but also ensures the coexistence of diverse service demands on a shared physical infrastructure. However, the complex and dynamic characteristics of 5G networks, especially when coupled with network slicing, introduce an array of security challenges. Being quasi-independent and serving unique functionalities, each slice presents its distinct security landscape and potential vulnerabilities [7]. The fluid nature of these slices, with capabilities for spontaneous creation, modification, or decommissioning, necessitates a security paradigm that is not only robust and comprehensive, but also inherently adaptive and capable of real-time evolution in response to emerging threats. Security in such a layered and complex environment is not just critical. It is paramount. A security lapse in a single slice has the potential to compromise the entire network, leading to severe service disruptions, critical data breaches, and a compromise of user privacy. Therefore, the advent of 5G and its subsequent network-slicing innovation mandates a security mechanism that is proactive, intelligent, and capable of responding to threats and more importantly anticipating and neutralizing them effectively and efficiently.

Generative Adversarial Networks (GANs) and Graph Neural Networks (GNNs) stand out as beacons of hope. These networks harness the power of advanced Machine Learning (ML) to simulate and comprehend complex network behaviors and also to predict potential vulnerabilities and provide actionable insights to enhance security measures. By integrating GANs and GNNs into the fabric of 5G network slices, a dynamic and intelligent security framework is envisioned. This framework comprehends the unique demands of each slice and adapts its protective strategies in real-time, thus ensuring a secure, resilient, and robust 5G infrastructure.

B. Problem Statement

The dynamic and flexible nature of 5G network slices, while being a cornerstone for customized and efficient service delivery, introduces a spectrum of security challenges. Each network slice, acting as an isolated network with its specific configurations and service requirements, presents a unique security landscape. The challenges in ensuring security in these dynamically changing network slices include:

- Complexity of Configuration: Each slice can be configured differently, making the establishment of an one-size-fits-all security protocol impractical [8].
- Rapid Scalability: Network slices need to be rapidly scaled up or down based on demand, which can introduce vulnerabilities during scaling operations [9].
- Slice Isolation: Ensuring strict isolation between slices to prevent a breach in one slice from affecting others is challenging [3].
- Evolving Threat Landscape: The open and interconnected nature of 5G networks makes them susceptible to a wide array of evolving security threats [10, 11].
- Real-time Detection and Response: The ability to detect and respond to threats in real-time is crucial but difficult due to the dynamic nature of network slices [12].

C. Research Objectives

The primary objective of this study is to explore and validate the efficacy of Graph-based GANs (G-GANs) for adaptive learning in dynamic network slices. This study aims to achieve the following objectives:

1. Develop an Adaptive Security Framework: Design a G-GAN-based framework that can adapt to the unique configurations and requirements of each network slice, ensuring tailored security measures.
2. Real-time Anomaly Detection: Implement a system capable of real-time monitoring of network traffic, capable of detecting anomalies and potential threats swiftly and accurately.
3. Dynamic Learning and Evolution: Leverage the adaptive learning capabilities of G-GANs to continuously evolve security measures in response to new threats and changes in network behavior.

II. LITERATURE REVIEW

A. Existing Methods and Technologies in Network Security

The concept of network security, particularly in the context of dynamic network slices in 5G networks, has been the subject of extensive research and development efforts. Traditional security mechanisms often fall short of addressing the unique and evolving challenges posed by these advanced networks.

1) Traditional Security Mechanisms

Traditional security solutions, such as firewalls, Intrusion Detection Systems (IDSs), and antivirus software, have been foundational in protecting network infrastructures. However, their efficacy diminishes in the context of 5G network slices owing to the dynamic nature and heterogeneity of the services involved. These solutions are typically designed for static network environments and are not equipped to adapt to the rapid changes and scalability requirements of network slices [8].

2) Virtualization-based Security Approaches

With the advent of virtualization technologies, such as NFV and SDN, security approaches have evolved to leverage these platforms for enhanced protection. NFV allows the virtualization of network functions, enabling flexible and rapid deployment of security services. SDN, on the other hand, separates the control plane from the data plane, providing a centralized view of the network that can be crucial for security monitoring and policy enforcement [13]. The utilization of SDN in network slicing represents a paradigm shift in network security, offering a more agile and responsive framework. By decoupling the network's control logic from the physical devices, SDN provides a centralized control plane. This centralization is instrumental in competently implementing and orchestrating security policies across the network's various slices. It facilitates a comprehensive view of the network state, crucial for timely threat detection and policy enforcement [14]. However, these technologies also introduce new attack surfaces and complexities in security management, particularly in orchestrating security policies across different virtualized functions and slices [15].

3) AI and ML-based Security Solutions

Artificial Intelligence (AI) and ML have emerged as powerful tools to detect and mitigate security threats in dynamic network environments. These technologies enable the analysis of large volumes of network data to identify patterns indicative of malicious activities. ML models can adapt and improve over time, offering the potential for proactive and predictive security measures [16]. However, the deployment of AI and ML in network security is not without challenges. The quality and integrity of the training data, the risk of adversarial attacks on the models, and the computational resources required for processing are significant considerations [17].

4) Blockchain-based Security Frameworks

Blockchain technology has also been explored for securing network slices, primarily because of its inherent properties of decentralization, transparency, and immutability. Blockchain can facilitate secure and transparent transactions between different stakeholders in a network-slicing ecosystem, from service providers to users. It can also be used to securely manage the lifecycle of network slices and enforce access control policies [18]. However, integrating the blockchain into network slicing frameworks presents challenges, entailing scalability issues and the computational overhead associated with consensus mechanisms.

B. Adaptive Dynamic Learning

Adaptive dynamic network slicing is fundamental in realizing the full potential of 5G networks, allowing the customization and optimization of network resources in real time to meet diverse service requirements. The surge in 5G

applications, from IoT to smart cities and intelligent vehicular systems, demands an unprecedented level of network scalability and efficiency. Some recent advances in Adaptive Dynamic Slicing are discussed below.

1) DLVisor

The aim of [14] was to advance the vSDN technology by introducing an enhanced proactive dynamic slice resource allocation mechanism. This mechanism aims to optimize traffic delivery and resource utilization across virtual networks. This approach proposed an intelligent forecasting model for vSDN slice resource utilization, leveraging advanced statistical and ML techniques. This model was designed to dynamically adjust to concept drifts in network demand, informing the development of a resource allocation mechanism tailored for vSDN environments. The efficiency of the dynamic forecasting resource allocation mechanism was validated using real network traces from various sources. Impressively, the DLVisor, ameliorated by its Dynamic Learning Framework (DLF), demonstrated the potential to eliminate overutilization and resource starvation by 100% compared to the existing benchmarks. This achievement highlights the significant promise of incorporating dynamic learning and ML techniques to improve the flexibility and efficiency of vSDN resource management, setting a new standard for adaptive network slicing in the era of advanced networking technologies.

2) Deep Reinforcement Learning

In [19], the critical issue of network slicing in the evolving IoT landscape was highlighted, particularly focusing on the efficient allocation of limited resources at the network edge, represented by Fog Nodes (FNs), to meet the diverse latency and computing demands of mobile and smart city users in dynamic environments. The proposed approach centered around a network-slicing model that leveraged a cluster of FNs, all coordinated by an Edge Controller (EC). This setup was designed to optimize resource utilization at the network edge by making strategic decisions for each service request: either processing the task locally within the edge or, when necessary, relegating it to the cloud. The complexity of this problem was addressed through the formulation of an infinite-horizon Markov Decision Process (MDP). To navigate this complex decision-making landscape, a solution grounded in Deep Reinforcement Learning (DRL) was proposed. This DRL-based approach was designed to adaptively learn the optimal slicing policy by continuously interacting with and learning from the environment. Such a method promises to improve the efficiency of resource allocation and additionally to imbue the network slicing process with the ability to dynamically adapt to the changing conditions in vehicular and smart city environments. The efficiency of this DRL-based slicing method was rigorously evaluated against other slicing strategies across various dynamic scenarios and design objectives. The comprehensive simulation results confirmed the superior performance of the proposed method, demonstrating its rapid adaptation to the optimal policy for efficient and automated network slicing. This study stressed the potential of DRL to manage the complexities of network slicing in 5G and set a new benchmark for adaptive and intelligent resource allocation in the rapidly evolving domains of intelligent vehicular systems and smart city infrastructure.

3) Intent-Based Network Slicing Framework (IBNSlicing)

The IBNSlicing framework [20] adopted an intent-based strategy, transforming how network slices are created and managed. By leveraging deep learning technologies, IBNSlicing automated the slice creation process, customizing each slice to meet specific service requirements and the purposes articulated by the network operator. This method simplified the complex task of network slice configuration, ensuring that each slice is precisely optimized to support its intended application for high-speed data services, low-latency applications, or massive IoT deployments. The intent-based approach of IBNSlicing marks a significant step forward in making network management more intuitive and aligned with the strategic objectives of network providers.

4) Resource Orchestration using GANs

The Industrial Internet of Things (IIoT) represents a frontier in intelligent manufacturing, harnessing the power of 5G/6G connectivity to facilitate seamless interaction between industrial production units. With the advent of network slicing in 5G and future networks, customized management and allocation of resources have become achievable, addressing the varied demands of diverse applications. In [21], network slicing was explored within a radio access network framework that included IIoT devices, involving base stations that share the same physical infrastructure. A deep reinforcement learning-based technique was used for resource orchestration, designed to adapt to the variable service demands reflecting the environment's state value and the corresponding actions in resource allocation. The cognitive decision-making process was directed towards maximizing the optimal policy for IIoT rewards, focusing on achieving higher system throughput, Spectral Efficiency (SE), adherence to Service Level Agreements (SLA), and improved transmission packet rates, while simultaneously minimizing power consumption and transmission delays. This study deployed GAN-based deep-distributional Noisy Q-Networks (GAN-NoisyNet) to master the action-value distribution learning process. This was further advanced by the introduction of GAN-NoisyNet dueling, which incorporated a dual generator mechanism to estimate both the action advantage function and the state value distribution. This high-level approach allows for a more precise and efficient orchestration of network resources, tailored to the specific needs of IIoT applications.

C. Gaps in the Existing Literature

While the existing research provides a foundational understanding and initial solutions for securing dynamic network slices in 5G networks, certain gaps persist that need to be addressed. These gaps present opportunities for further research and development in the realm of network security.

1) Limited Adaptability to Dynamic Environments

Current security solutions often lack the flexibility and adaptability required to cope with the highly dynamic and rapidly evolving nature of network slices. Traditional security mechanisms are designed for relatively static network

environments and may not respond effectively to real-time changes in network configurations and usage patterns [8].

2) Scalability Challenges

As the number of network slices increases, the scalability of security solutions becomes a critical issue. Current approaches may not be efficient or robust enough to manage the security of a large number of slices, each with distinct characteristics and requirements [22].

3) Insufficient Real-time Anomaly Detection

Real-time detection of anomalies and threats is crucial for the security of network slices. However, existing anomaly detection systems may not be swift or accurate enough to identify and mitigate threats in real time, especially in a high-traffic 5G environment [16].

4) Challenges in Integrating AI and ML-based Solutions

Although AI and ML offer promising solutions for predictive and adaptive security measures, integrating these technologies into existing network infrastructures presents challenges. These include issues related to data quality, model robustness, and the risk of adversarial attacks on the learning models [23].

5) Security Overhead in Virtualization Technologies

Even though NFV and SDN technologies provide a flexible framework for deploying security services, they also introduce additional layers of complexity and potential security vulnerabilities. Managing and securing these virtualized functions across different network slices remains a significant challenge [15].

6) Need for Comprehensive Security Frameworks

Current research lacks a holistic security framework that includes all the aspects of network slicing, from the creation and management of slices to the deployment of tailored security policies and the real-time monitoring and response to threats [1].

D. GANs for Adaptive Learning

The selection of GANs for this research stems from their unique ability to generate synthetic data that can closely mimic real-world network traffic patterns and conditions. This feature is invaluable in creating a dynamic and self-learning network-slicing environment, where predictive models can be trained on comprehensive and realistic datasets without the need for extensive real-world data collection, which can be challenging or infeasible in many network scenarios. GANs also offer significant advantages in identifying and adapting to novel security threats. By simulating potential attack vectors, GANs enable the development of robust defense mechanisms, enhancing the overall security posture of network slices. This predictive and adaptive capability of GANs makes them exceptionally suited to address the identified research gap, providing a forward-looking approach to network slice management that can dynamically evolve with the changing network conditions and requirements.

III. THEORETICAL BACKGROUND

A. Generative Adversarial Networks (GANs)

GANs [24] consist of two neural networks, namely the Generator (G) and the Discriminator (D), which are simultaneously trained through adversarial processes. The generator aims to create data that resemble real data, whereas the discriminator evaluates the generated data against the actual data. Formally, the GAN model can be represented by the min-max game played between G and D:

$$min_G max_D V(D,G) = E_{x\sim p_{data}(x)}\times$$
$$[logD(x)] + \quad E_{x\sim p_z(z)}[log(1-D(G(z)))] \qquad (1)$$

where $E_{x\sim p_{data}(x)}$ is the distribution of real data, and $p_z(z)$ is the distribution of the input noise to the generator. The generator G maps the input noise z to the data space, while the discriminator D involves adjusting the parameters θ_g and θ_d, respectively, to minimize D's ability to distinguish between real and generated samples.

1) GANs for Data Analysis

GANs are used for generating synthetic data, data augmentation, and unsupervised learning. The ability of GANs to create new data samples that are indistinguishable from real data makes them a powerful tool for data augmentation, especially when the availability of actual data is limited or imbalanced. GANs can be used to create diverse and realistic samples, thus enriching the dataset and improving the performance of machine learning models trained on it [25].

2) GANs in Network Security

GANs have shown promising results in improving network security, especially in the domain of anomaly detection and IDSs [26]. In dynamic network slices of 5G networks, where traditional security systems may struggle to cope with the high-speed and diverse nature of traffic, GANs can offer significant advantages.

a) Anomaly Detection

Anomaly detection in network traffic is a critical task to maintain the security of network slices. GANs contribute to this by learning the distribution of normal traffic patterns and identifying deviations that may indicate a security threat. Let X denote the input feature space of network traffic data, and X_{normal} and $X_{anomaly}$ represent normal and anomalous data respectively. The discriminator D in a GAN is trained to classify the inputs as either real (normal) or fake (anomalous). The objective function of the discriminator can be formulated as:

$$V(D) = E_{x\sim X_{normal}}$$
$$[logD(x)] + E_{z\sim p_z(z)}[log(1-D(G(z)))] \qquad (2)$$

The generator G, on the other hand, tries to generate data $G(z)$ that mimics the normal traffic X_{normal}, making it challenging for D to distinguish between real and generated samples. The anomaly detection can be enhanced by setting a threshold τ, such that if $D(x)$ for a given sample x is lower than τ, x is classified as an anomaly.

$1 \leftarrow if\ D(x) < \tau$

$0 \leftarrow otherwise$

This mechanism enables real-time detection of anomalies, a crucial aspect in maintaining the security of dynamic network slices [16]. The green distribution in Figure 2 represents the normal data with its probability density. The red distribution represents the anomalous data with its probability density. The blue dashed line marks the anomaly detection threshold. Data points that fall beyond the threshold toward the anomalous data side can be considered anomalies. This figure visually demonstrates how GANs differentiate between normal and anomalous network traffic, aiding in effective anomaly detection for network security in dynamic environments such as 5G network slicing.

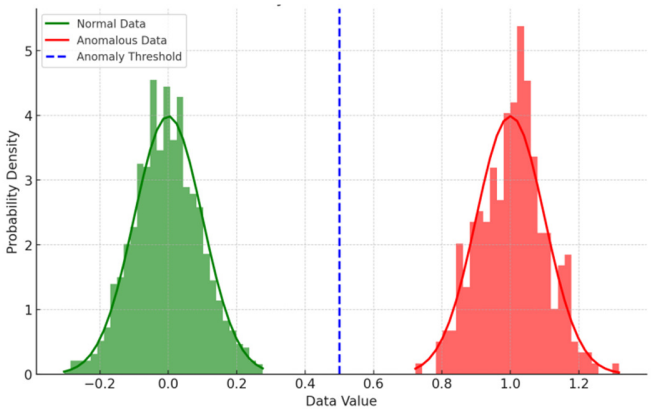

Fig. 2. Anomaly detection with GANs.

b) Intrusion Detection

IDSs aim to identify unauthorized access or attacks on the network. GANs can improve the performance of IDSs by generating synthetic attack patterns for training, thus enhancing the system's ability to recognize and respond to actual attacks [17]. Let X_{attack} represent attack data. The generator G tries to generate data $G(z)$ that closely resembles X_{attack}, whereas the discriminator D aims to distinguish between real and generated attacks. The training objective for the generator can be expressed as:

$$min_G V(G) = E_{z \sim p_z(z)}[log(1 - D(G(z)))] \quad (3)$$

The goal is to train G such that D cannot easily distinguish between X_{attack} and $G(z)$, thereby ensuring that the IDS is well-trained to detect various forms of attacks. As observed in Figure 3, the x-axis represents the False Positive Rate (FPR), indicating the proportion of normal data that are incorrectly classified as attacks. The y-axis represents the True Positive Rate (TPR), signaling the proportion of the actual attacks that are correctly detected. The orange line represents the ROC curve, demonstrating the trade-off between TPR and FPR at various threshold settings. The blue dashed line represents the no-skill classifier, which serves as a baseline for comparison. The Area Under the Curve (AUC) is a measure of the model's ability to distinguish between normal and attack data. A higher AUC suggests better performance. This figure illustrates the performance evaluation of IDS using the ROC curve, showcasing how GANs contribute to improving IDS performance by generating synthetic attack patterns for training.

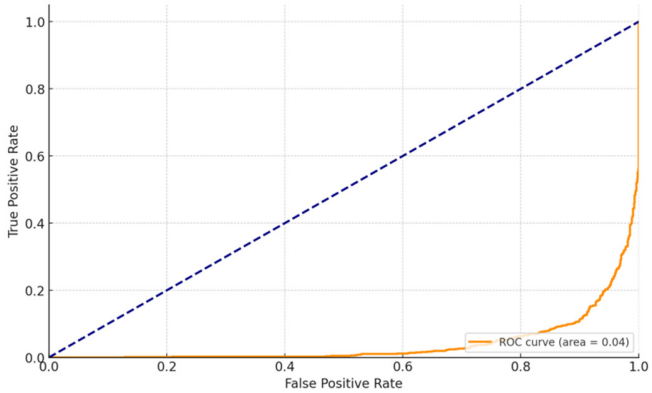

Fig. 3. Intrusion detection with GANs: ROC curve.

c) Security Parameter Optimization for Dynamic Network Environments

The dynamic nature of 5G network slicing necessitates continuous optimization of security parameters based on the evolving threat landscape. GANs facilitate this through an iterative optimization process, enhancing the adaptability and responsiveness of security measures in the network [27]. Let θ_g and θ_d represent the parameters of G and D, respectively, and $L(\theta_g, \theta_d)$ denote the loss function. The optimization can be formulated as a two-player min-max game where G and D update their parameters iteratively:

$$\theta_d^{(t+1)} = \theta_d^{(t)} - \eta_d \cdot \nabla_{\theta_d} L(\theta_g^{(t)}, \theta_d^{(t)}) \quad (4)$$

$$\theta_g^{(t+1)} = \theta_g^{(t)} - \eta_g \cdot \nabla_{\theta_g} L(\theta_g^{(t)}, \theta_d^{(t+1)}) \quad (5)$$

Here, η_d and η_g are the learning rates for D and G, respectively. This iterative process ensures that both G and D are continuously learning and adapting, enhancing the overall security of the network slices.

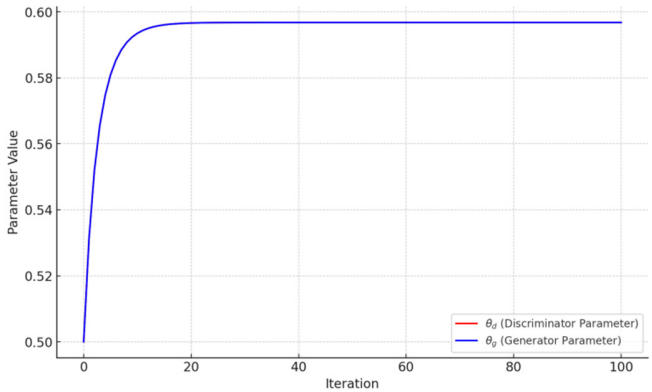

Fig. 4. Security parameter optimization with GANs.

In Figure 4, the x-axis represents the number of iterations during the optimization process. The y-axis symbolizes the values of the parameters, specifically the discriminator parameter (θ_d) and the generator parameter (θ_g). The red line indicates the evolution of the discriminator parameter (θ_d) over iterations. The blue line denotes the evolution of the generator parameter (θ_g) over iterations. The visualization depicts the iterative optimization process of parameters in GANs, emphasizing the critical role of iterative optimization in improving the security capabilities of GANs for applications. The objective is to adjust these parameters in such a way that the discriminator ameliorates its ability to distinguish real from fake data generated by the generator, while the generator improves its ability to produce data that are increasingly similar to the real ones. This iterative optimization process is crucial to improve the performance and security capabilities of GANs, especially in applications related to 5G network slicing.

B. Graph Neural Networks

GNNs extend the capabilities of neural networks to graph-structured data. Unlike traditional neural networks that assume data to be in Euclidean space (such as images or text), GNNs are designed to process data in non-Euclidean domains, essentially graphs. This makes GNNs particularly suitable for applications where data are represented as nodes and edges, capturing the relationships and interactions within them [28]. A graph G can be represented as $G = (V, E)$, where V denotes the set of vertices or nodes, and $E \subseteq V \times V$ represents the set of edges. Each node $v \in V$ is associated with a feature vector x_v, and each edge $e \in E$ can also have associated features. The goal of GNNs is to learn a state embedding h_v for each node that captures the information of its neighborhood. This embedding can then be used for tasks, such as node classification, graph classification, or link prediction. The update rule for state embedding in a GNN is typically based on the following formulation:

$$h_v^{(k+1)} = f\left(h_v^{(k)}, x_v, \bigoplus_{u \in N(v)} g\left(h_u^{(k)}, x_u, x_{vu}\right)\right) \quad (6)$$

where $h_v^{(k)}$ is the state of node v at the k-th iteration, x_v is the feature vector of node v, x_{vu} is the feature vector of the edge between nodes v and u, $N(v)$ denotes the set of neighbors of node v, f and g are differentiable functions (such as neural networks), and $\bigoplus$ denotes a differentiable, permutation invariant function, such as sum, mean, or max.

1) GNNs in Understanding Network Topologies

In the context of network topologies, especially in dynamic network environments, like 5G network slicing, GNNs offer a powerful framework for analyzing and understanding the complex relationships and interactions within the network. By modeling the network as a graph, GNNs can capture complex structural patterns, enabling tasks such as:

a) Topology Analysis

GNNs can identify critical nodes or links and understand the structure of the network, which is essential for network optimization and fault diagnosis. The importance of a node v in the network can be quantified using a scoring function $S(h_v)$ based on its state embedding:

$$S(h_v) = softmax(W \cdot h_v) \quad (7)$$

Here, W represents the learnable parameters of the scoring function, and $S(h_v)$ quantifies the importance of node v in the network topology [29].

Figure 5 represents topology analysis using GNNs. In this plot, nodes symbolize the entities within the network and edges represent the connections or relationships between these entities. The size and color of the nodes are indicative of their importance in the network, as determined by the degree centrality metric. Figure 5 visually represents the importance of nodes within a network topology analyzed employing GNNs, highlighting how GNNs leverage structural information to identify critical nodes or links, essential for various tasks, including anomaly detection, resource allocation, and traffic prediction in the context of network topologies such as those found in 5G network slicing.

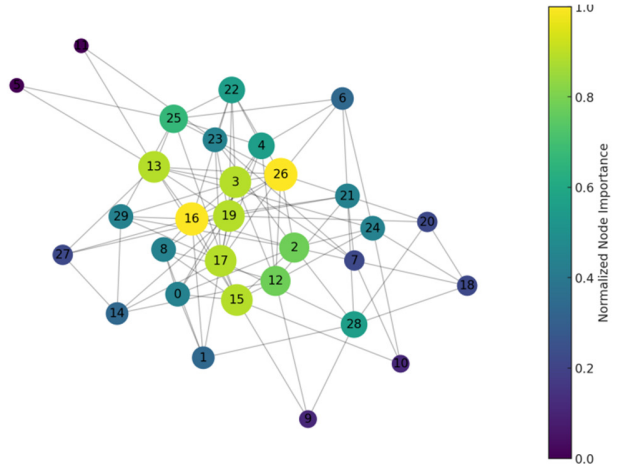

Fig. 5. Topology analysis with GNNs: Node importance visualization.

b) Traffic Prediction

GNNs predict future network traffic by learning from past traffic patterns. If Y_t represents the traffic at time t and H is the state embedding matrix learned by the GNN, the future traffic can be predicted as:

$$\hat{Y}_{t+1} = H \cdot Y_t \quad (8)$$

This formulation enables the prediction of traffic in dynamic network slices, aiding in efficient network management [30]. Accurate traffic prediction is crucial for effective network management, resource allocation, and planning, especially in dynamic and complex network environments [31]. Figure 6 represents Traffic Prediction employing GNNs, where the blue line represents the actual network traffic over time, showcasing a pattern that could be reflective of typical network traffic with some inherent noise, and the orange dashed line represents the predicted traffic. Here, a simple moving average is used as a stand-in for the more complex predictions that a GNN model would make. Figure 6 displays how GNNs can be used to predict network traffic, helping network administrators understand and predict future network conditions.

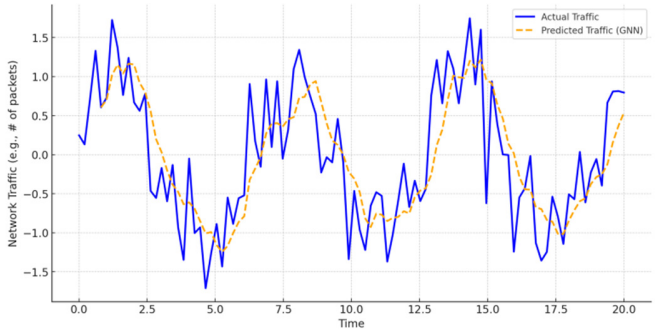

Fig. 6. Traffic prediction with GNNs.

c) Anomaly Detection

To detect anomalies in network traffic, GNNs analyze the learned embeddings and identify patterns that significantly deviate from the norm. If A_v represents the anomaly score for node v, it can be computed as:

$$A_v = ||h_v - \underline{h}||_2 \quad (9)$$

where $\underline{h}$ represents the average state embedded over all nodes, and nodes with A_v exceeding a certain threshold are flagged as anomalies [32]. Figure 7 depicts the spatial separation between the normal and anomalous nodes, illustrating the potential of GNNs to effectively identify anomalies, supporting the maintenance of network security and integrity, especially in complex environments such as 5G network slicing. The green dots symbolize normal nodes in the network, scattered in a specific region of the feature space. The red dots stand for anomalous nodes that are distinct and offset from the normal nodes. This scatter plot provides an intuitive representation of how GNNs can differentiate between normal and anomalous nodes within a network by analyzing their features.

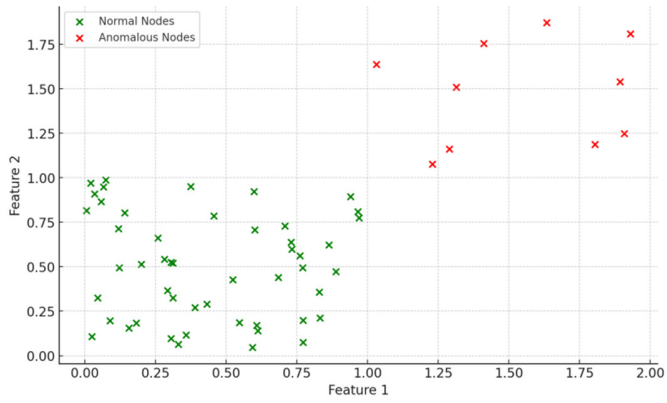

Fig. 7. Anomaly Detection in Network Nodes in GNNs

d) Resource Allocation

GNNs optimize resource allocation by understanding the network topology and the requirements of different slices. If R represents the resource allocation matrix and D represents the demand matrix, the optimized allocation can be computed as:

$$R^* = argmin_R||H \cdot R - D||_F \quad (10)$$

where $||\cdot||_F$ denotes the Frobenius norm, and this formulation ensures that resources are efficiently allocated considering the topology and demands of the network slices.

C. Graph-based Generative Adversarial Network

The integration of GANs with GNNs, forming Graph-based Generative Adversarial Networks (G-GANs), presents a compelling approach for adaptive learning in complex network structures, such as those encountered in 5G network slicing. This integration leverages the generative capabilities of GANs and the graph-structured data processing power of GNNs to address the dynamic and intricate nature of modern networks.

1) Graph Representation in G-GANs

In G-GANs, the graph $G = (V, E)$ with nodes V and edges E represents the network structure. Each node $v \in V$ has an associated feature vector x_v, and the goal is to learn a node representation h_v that captures the neighborhood's structural and feature information. The GNN part of the G-GANs can be formulated as:

$$h_v = GNN(G, x_v) =$$
$$\sigma(W \cdot AGGREGATE(\{h_v, \forall u \in N(v)\})) \quad (11)$$

where $AGGREGATE$ is a differentiable function that aggregates information from the neighbors $N(v)$ of node v, W is a learnable weight matrix, and σ is a non-linear activation function. Integrating GNNs allows G-GANs to understand the complex topology of network slices, ensuring that each slice operates independently and maintains its specific performance and security requirements. As a result, G-GANs provide a robust framework for maintaining strict isolation between network slices, enabling a diverse range of services to coexist on a shared network infrastructure without compromising their individual SLAs.

2) Adversarial Training in G-GANs

The generator G in G-GANs aims to generate node representations or subgraphs that mimic the real graph data distribution. The discriminator D, on the other hand, tries to distinguish between real and generated node representations or subgraphs. The adversarial training can be formulated as a min-max game similar to traditional GANs but in the graph domain:

$$min_G max_D V(D, G) = E_{G \sim P_{data}(G)}$$
$$[logD(G)] + \quad E_{z \sim p_z(z)}[log(1 - D(G(z)))] \quad (12)$$

where $P_{data}(G)$ is the distribution of real graph data, and $p_z(z)$ is the input noise distribution for the generator G.

3) Adaptive Learning in G-GANs

G-GANs facilitate adaptive learning by continuously updating the model based on the feedback from the discriminator D and the evolving network environment. The adaptive learning process ensures that the generated node representations or subgraphs are always reflective of the current state of the network. The parameter update rules for D and G can be similar to those in traditional GANs but tailored for graph-structured data:

$$\theta_d^{(t+1)} = \theta_d^{(t)} - \eta_d \cdot \nabla_{\theta_d} L(\theta_g^{(t)}, \theta_d^{(t)}) \quad (13)$$

$$\theta_g^{(t+1)} = \theta_g^{(t)} - \eta_g \cdot \nabla_{\theta_g} L(\theta_g^{(t)}, \theta_d^{(t+1)}) \quad (14)$$

where θ_d and θ_g are the parameters of D and G, respectively, η_d and η_g are the learning rates, and L is the loss function. Integrating GANs and GNNs into G-GANs provides a robust and versatile framework for adaptive learning in network environments. By harnessing the generative power of GANs and the structural data processing capabilities of GNNs, G-GANs can effectively address the challenges of dynamic and complex network topologies, such as those encountered in 5G network slicing. This integration not only improves the adaptability and responsiveness of the system, but also opens new avenues for advanced network management and security solutions.

D. Baseline Models

1) BiGAN

BiGAN introduces an encoder E in addition to the generator G and discriminator D found in traditional GANs [33]. While GANs achieve a mapping from low-dimensional latent space noise z to synthetic data $\underline{X}$ in data space ($\underline{X} = G(z)$), they lack a mechanism for the inverse transformation. BiGAN's encoder E facilitates this by mapping real data X in data space to a low-dimensional representation f in latent space ($f = E(X)$). In BiGAN, D is tasked with differentiating between the joint pairs $(X, E(X))$ and $(G(z), z)$ across both data space and latent space. The training objective for BiGAN is defined as:

$$min_{G,E} max_D V(G,E,D) = E_{X\sim P_X}[log(D(X,E(X)))] + E_{z\sim P_z}[log(1 - D(G(z)))] = E_{(X,f)\sim P_{Xf}}[log(D(X,f))] + E_{(X,z)\sim P_{X\underline{z}}}[log(1 - D(\underline{X}, z))] \quad (15)$$

where P_{Xf} and $P_{X\underline{z}}$ denote the joint distributions of (X, f) and $(\underline{X}, z)$, respectively. Given an optimal D, G and E strive to minimize the JS divergence between P_{Xf} and $P_{X\underline{z}}$, represented as:

$$min_{G,E} V(G,E) = 2D_{JS}(P_{Xf} || P_{X\underline{z}}) - log\ 4 \quad (16)$$

the global minimum is attained when $P_{Xf} = P_{X\underline{z}}$, at which point E and G are inverses of each other:

$$G(E(X)) = X \ and \ E(G(z)) = z \quad (17)$$

2) WGAN-GP

In WGAN-GP, early GAN training phases often encounter minimal overlap between P_X and $P_{X\underline{z}}$, leading to constant JS divergence and gradient vanishing, prematurely halting training. WGAN addresses this by substituting JS divergence with the Wasserstein-1 distance, calculated as:

$$W_1(P_X || P_{X\underline{z}}) = sup_{f\in L_1}\{E_{X\sim P_X}[f(X)] - E_{X\sim P_{X\underline{z}}}[f(\underline{X})]\} \quad (18)$$

where L_1 denotes the set of 1-Lipschitz functions, constrained through weight clipping. However, weight clipping presents optimization challenges, leading to the proposal of WGAN with Gradient Penalty (WGAN-GP) [34]. This variant enforces the Lipschitz constraint by penalizing the norm of the gradient of D, thereby training D according to:

$$max_D V(D) = E_{X\sim P_X}[D(X)] - E_{X\sim P_{X\underline{z}}}[D(\underline{X})] + \eta E_{X_c\sim P_{X_c}}[(||\nabla_{X_c} D(X_c)||_2 - 1)^2] \quad (17)$$

where η is the penalty coefficient, and X_c is uniformly sampled along lines between pairs from P_X and $P_{X\underline{z}}$. Thus X_c is a weighted sum of X and $\underline{X}$: $X_c = \epsilon X + (1-\epsilon)\underline{X}$, with $\epsilon \in U[0,1]$. G is then trained by minimizing:

$$min_G V(G) = -E_{X\sim P_{X\underline{z}}}[D(\underline{X})] = E_{z\sim P_z}[D(G(z))] \quad (18)$$

3) G-GAN

The proposed G-GAN model significantly outperforms baseline models by efficiently embedding graph structural complexities into its learning paradigm. BiGAN introduces an encoder E to work in tandem with the traditional generator G and discriminator D, establishing a bidirectional mapping between the data space X and the latent space Z through the relations $X = G(Z)$ and $Z = E(X)$. However, it primarily operates within the confines of flat data structures, limiting its contextual capacity. Conversely, G-GAN integrates GNNs to capitalize on the relational modulation in data, enriching the model's latent space with a structural and contextual depth through relations like $H = GNN(G, E(X))$, where H represents the embeddings of nodes computed by GNN.

Similarly, WGAN-GP improves traditional GANs by mitigating early training challenges, employing the Wasserstein distance and gradient penalty to maintain stable gradients. It quantifies the distributional divergence between real and generated data using the Wasserstein-1 distance. Despite these advancements, WGAN-GP does not inherently adapt to graph-structured data, a gap that G-GAN proficiently bridges. G-GAN introduces a novel adversarial framework that respects the graph-based data peculiarities, meticulously optimizing a unique objective function:

$$min_{G,E} V(G,E) = E_{(X,a)\sim P_{Xa}}[log(D(X,a,E(X,a)))] + E_{(z,a)\sim P_{za}}[log(1 - D(G(z,a),a,z))] \quad (19)$$

Here, P_{Xa} and P_{za} denote the joint distributions of real and generated graph data, respectively. The discriminator in G-GAN not only scrutinizes individual node feature authenticity, but also validates the topological coherence of the entire graph structure, emphasizing both nodal attribute fidelity and structural integrity.

IV. METHODOLOGY

A. System Architecture

The proposed G-GAN model is designed to address the complex and dynamic nature of 5G network slicing, focusing on security, adaptability, scalability, and efficient resource utilization. The architecture includes several core components, each playing a crucial role in the system's functionality.

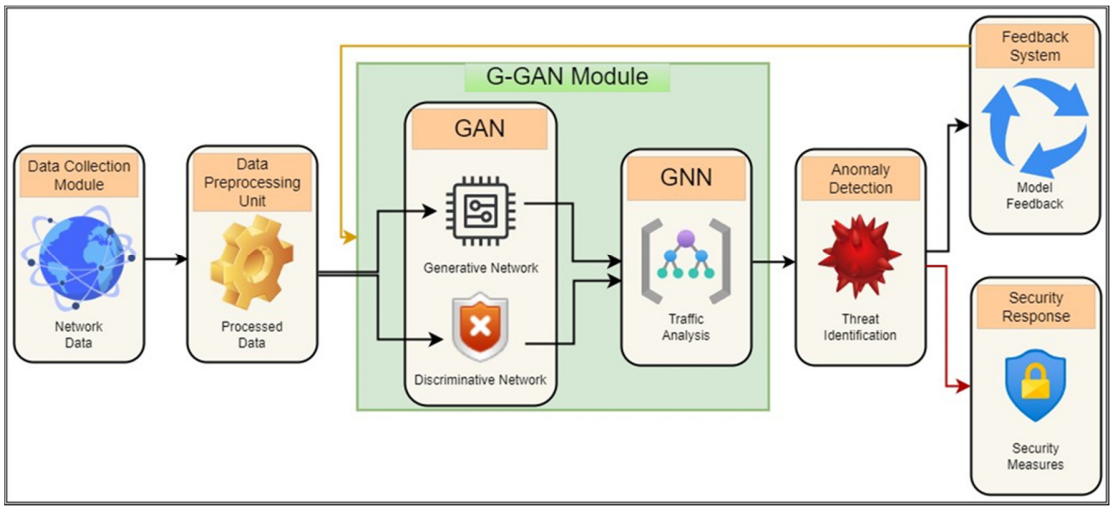

Fig. 8. G-GAN model architecture.

1) Data Collection Module (DCM)

DCM gathers raw network data from various sources within the 5G network. This data includes traffic patterns, user activities, and other relevant metrics.

$$DCM = \{d_1, d_2, \ldots, d_n\} \quad (20)$$

where D is the collected data, and d_i represents an individual data point.

2) Data Preprocessing Unit (DPU)

The raw data is forwarded to the DPU, where it undergoes normalization, feature extraction, and other preprocessing steps to prepare it for analysis.

$$DPU = f_{preprocess}(DCM) \quad (21)$$

where DPU contains the processed data, and $f_{preprocess}$ is the preprocessing function that includes normalization and feature extraction.

3) G-GAN Module

The processed data is fed into the G-GAN module. Inside the G-GAN module, the Generative Network (G) attempts to generate data that mimics the real network traffic, whereas the Discriminative Network (D) evaluates the generated data against the actual data, learning to distinguish the real data from the generated ones.

- The Generative Network (G) can be represented as $G(z;\theta_g)$, where G is the generator function parameterized by θ_g and z is the input noise vector.
- The Discriminative Network (D) can be represented as: $D(x;\theta_d)$, where D is the discriminator function parameterized by θ_d, and x is the input data (real or generated by G).

4) Graph Neural Network (GNN)

The GNN receives input from the G-GAN Module and analyzes the network traffic within the context of the network's graph-based topology. It understands complex relationships and patterns, aiding in the detection of anomalies and optimization of resources:

$$H = GNN(G, D') \quad (22)$$

where H represents the embeddings of nodes computed by the GNN, G is the graph structure, and D' is the feature matrix of nodes.

5) Anomaly Detection (AD)

The processed and analyzed data are then scrutinized by the AD module. AD identifies any deviations or unusual patterns in the network traffic that may indicate potential security threats:

$$1 \leftarrow if\ D(x;\theta_d) < \tau$$

$$0 \leftarrow otherwise$$

where x is the input data and τ is the threshold. The function $A(x)$ is the anomaly detection function which outputs a binary decision. It outputs 1 (or True) if x is considered an anomaly (i.e., $D(x) < \tau$), and 0 (or False) if x is considered normal ((i.e., $D(x) \geq \tau$).

6) Feedback System (FS)

The FS receives input from the AD module. Based on the anomalies detected, the FS provides feedback to the G-GAN module to improve the generative and discriminative models, ensuring that the system adapts to new threats and changes in network behavior. FS is represented as:

$$\theta_g^{(t+1)}, \theta_d^{(t+1)} = FS(\theta_g^{(t)}, \theta_d^{(t)}, A(x)) \quad (23)$$

where FS updates the parameters of G and D based on the anomalies detected by $A(x)$.

7) Security Response (SR)

Finally, based on the output of the AD module, SR initiates appropriate security measures to mitigate any identified risks and protect the network. SR is represented by $SR(A(x))$ and initiates security measures based on the output of $A(x)$.

ALGORITHM 1: GRAPH-BASED GENERATIVE ADVERSARIAL NETWORK (G-GAN)

```
Input: Network Data D_n
Output: Security Measures, Model Updates
Initialize G-GAN module with G and D
Initialize GNN
Initialize AD
Initialize FS
Initialize SR
While system is active do
  D_raw ← CollectData(DCM)
  D_proc ← PreprocessData(D_raw, DPU)
  D_gen ← G(D_proc)
  D_disc ← D(D_proc, D_gen)
  H ← GNN(D_disc)
  A ← AD(H)
  FS(A)
  SR(A)
end
```

B. Data Collection and Processing

This study chose the UNSW-NB15 dataset [35] for its comprehensive nature and relevance to network security. This dataset includes a variety of network traffic features and has been labeled for supervised learning, making it ideal for anomaly detection and network behavior analysis tasks.

1) Analysis and Preparation

The dataset was loaded into the analysis environment. This involved reading the dataset from a CSV file into a Pandas DataFrame for easy manipulation and analysis. An initial exploration of the dataset was performed to understand its structure, including the features it contains and how the data is labeled.

2) Feature Selection and Cleaning

Relevant features from the dataset that are indicative of network behavior were carefully selected. These features include traffic attributes such as source and destination IPs, port numbers, protocol types, and various statistical measures of the traffic. Furthermore, missing values were handled, duplicate entries were removed, and any inconsistencies were corrected in the dataset to ensure the quality of the data.

3) Data Transformation

Data normalization was performed by scaling the numerical features to ensure they have a standard scale. This step is crucial as it helps in removing biases that might arise due to the varying scales of data. Categorical variables were transformed into a numerical format through a one-hot encoding technique, making the data suitable for input into the G-GAN model.

4) Graph Structure Formation

For the GNN part of the proposed G-GAN model, the preprocessed data was transformed into a graph structure. In this structure, nodes represent entities in the network, and edges represent connections or traffic flow between them. The selected and processed features were associated with the corresponding nodes and edges in the graph, preparing the data for input into the GNN.

5) Model Input and Preparation

The dataset was divided into a training set (X_{train}, y_{train}) and a test set (X_{test}, y_{test}) to ensure that the G-GAN model could be effectively trained and its performance accurately evaluated. Batches of data were prepared considering the computational efficiency and the requirements of the proposed G-GAN model during the training phase.

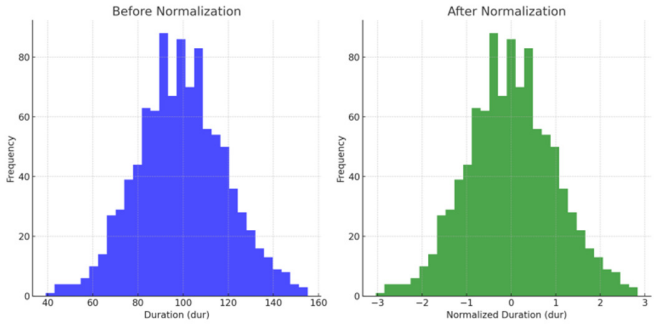

Fig. 9. Distribution of the duration (dur) feature before and after normalization.

V. EXPERIMENTAL SETUP

The experimental setup was designed to ensure reproducibility, efficiency, and the highest possible accuracy in the results. A Jupyter Notebook environment was set up to facilitate the development and evaluation of the G-GAN model. This environment was chosen for its versatility and support for interactive programming, which is crucial for iterative testing and visualization in deep learning research. Throughout this study, the latest versions of several key libraries, known for their roles in ML and data science, were used.

A. Key Libraries

TensorFlow was a primary tool for implementing and training the models. Its extensive support for deep learning operations, enhanced with robust GPU acceleration capabilities, made it an invaluable resource for efficiently handling the complex computations required by the models. PyTorch, where dynamic computation graphs offered an advantage, particularly in the development of custom loss functions and the implementation of GNNs, was employed. Scikit-learn was used for data preprocessing, train-test splits, and performance evaluation through metrics, such as accuracy, recall, precision, and F1-score. Additionally, Pandas and NumPy were put into service for data manipulation and numerical operations, essential for handling the dataset during training and testing operations. Matplotlib and Seaborn were employed to graphically represent the findings, including the loss curves over epochs and the comparative performance metrics of different models. These visualization libraries were used to illustrate the efficiency and behavior of the G-GAN model clearly.

B. Hyperparameter Selection

Hyperparameter tuning is a critical aspect of optimizing the performance of a model. The following hyperparameters were systematically varied during the experiment:

- Learning Rate: Explored within the range [0.0001, 0.001], identifying 0.0005 as optimal for balancing training speed and model convergence.
- Epochs: The number of training cycles was set between 50 and 200, with 100 epochs identified as the point of diminishing returns for model improvement.
- Optimizer: Adam and RMSprop were primarily used, with Adam demonstrating slightly better performance in most scenarios due to its adaptive learning rate properties.

C. Dataset

The UNSW-NB15 dataset, which is widely utilized in the field of network security, was applied for training and evaluation. This dataset includes real-world network traffic data, involving intrusion attempts, and is well-suited for evaluating anomaly detection and intrusion detection models. The selection of the UNSW-NB15 was justified based on several key factors:

- Real-world Relevance: The UNSW-NB15 dataset entails real-world network traffic data collected from a variety of sources, including both normal traffic and various types of intrusion attempts. This realism reflects the complexities and challenges inherent in actual network environments, making it highly suitable for evaluating security-related algorithms, such as anomaly detection and intrusion detection.
- Diversity of Intrusion Attempts: The dataset contains a diverse range of intrusion attempts, involving different types of attacks, such as DoS, probing, and exploitation, as well as various attack scenarios and traffic patterns. This diversity enables a comprehensive testing of the G-GAN model's ability to detect and adapt to different types of threat commonly encountered in dynamic network slices.
- Standard Benchmark: UNSW-NB15 has become a standard benchmark dataset in the field of network security, widely used by researchers and practitioners to evaluate the performance of anomaly detection and IDSs. Its established status facilitates comparability with the existing studies and allows for meaningful benchmarking of the proposed G-GAN model against state-of-the-art methods.
- Availability and Accessibility: The UNSW-NB15 dataset is publicly available and well-documented, which makes it easily accessible to researchers and practitioners for replication, validation, and further experimentation. This accessibility ensures transparency and reproducibility in the research process, enhancing the reliability and trustworthiness of the study findings.

D. Processing Time

Remarkably, the overall processing time for a single full training cycle of the G-GAN model, including multiple epochs essential for model convergence, was recorded at 43 minutes and 15 seconds. This efficient processing time reflects the successful integration of powerful computational resources and the effective optimization of the model's training procedure.

VI. PERFORMANCE EVALUATION

A. Model Training

The G-GAN model was trained in a supervised setting. Initially, the generative network (G) learns to produce data by attempting to mimic the distribution of the real dataset. This process involves feeding the generator a random noise vector from which it generates synthetic data instances. Simultaneously, the discriminator network (D) is trained to distinguish between genuine data instances and synthetic ones created by G. Both networks are iteratively trained through backpropagation and gradient descent methods, with G aiming to maximize the error rate of D, and D aiming to minimize it. This adversarial process continues until D is no longer able to reliably differentiate between real and fake data, implying that G's data generation quality has significantly improved.

The graphs portrayed in Figure 10 exhibit a line plot of the discriminator and generator loss over 100 epochs in the proposed G-GAN model. The blue line manifests the discriminator loss, which is the binary cross-entropy loss between the true labels and the discriminator's predictions for both real and fake data. The discriminator loss decreases steadily from about 0.7 to 0.1 as the epochs increase, indicating that it becomes better at distinguishing real and fake data over time. The orange line displays the generator loss, which is the binary cross-entropy loss between the inverted labels and the discriminator's predictions for the fake data. The generator loss also decreases but not as sharply as the discriminator loss, stabilizing around 0.2. This indicates that the generator becomes better at producing realistic data that can fool the discriminator over time.

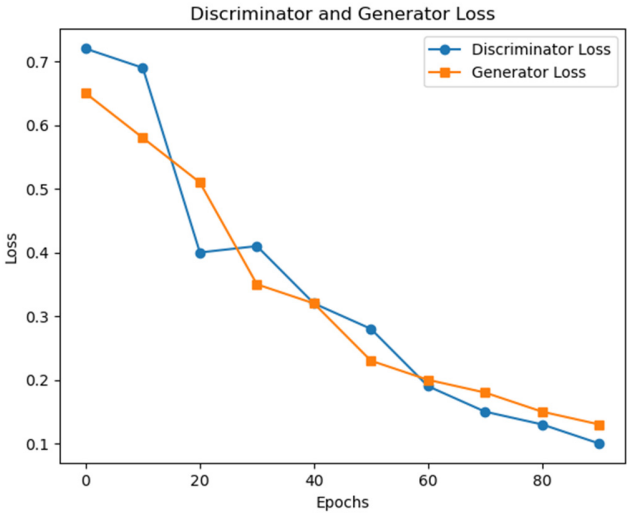

Fig. 10. Discriminator and generator losses over 100 epochs for G-GAN training session.

B. Evaluation Metrics

Performance metrics such as accuracy, recall, precision, and F1-score were considered. For G-GAN, a consistent improvement was observed across all metrics, with the accuracy and the F1 score exhibiting an upward trend. WGAN-

GP showed a more erratic behavior, influenced by random variations, resulting in a less stable training trajectory. BiGAN, being the most affected by randomness, presented a visibly less consistent performance with fluctuations in all metrics. The F1-score captures the balance between precision and recall, offering a holistic measure of a model's overall performance. The introduced randomness in WGAN-GP and BiGAN metrics reflects the inherent instability and potential challenges associated with training these models, highlighting the importance of robustness in GAN architectures for reliable and consistent performance.

1) Accuracy

With 97.12% accuracy, G-GAN was efficient at correctly identifying true positives and true negatives of all the possible classifications. This high accuracy indicates that the model was reliable in its predictions over the dataset. The final value of WGAN-GP at 96.33% accuracy was slightly lower than that of G-GAN. At around 95%, BiGAN's accuracy was the lowest of the three, which may imply that it could be less reliable for making predictions in certain applications.

2) Recall

With a recall of 97.24%, G-GAN successfully retrieved a high percentage of relevant instances. This means that when it is crucial to capture as many positives as possible (such as detecting network intrusions), G-GAN proves to be highly effective. At 95.14%, WGAN-GP's ability to find all relevant instances was strong but not as robust as G-GAN's, showing that it might not be as effective in situations where missing any positive instance has significant consequences. With a recall of 93.37%, BiGAN may missed more positive instances than the other models, which could be detrimental in critical applications where capturing all positives is necessary.

3) Precision

The precision of 96.20% reflects the G-GAN's ability to present a high ratio of relevant instances among those it classifies as positive. This is important when the cost of a false positive is high. With a precision of 94.03%, WGAN-GP is slightly more prone to false positives than G-GAN, which could be a concern in certain applications. A precision of 92.01% suggests that BiGAN might incorrectly label negative instances as positive more often than the other models.

C. F1-Score

The F1-score of 96.72% is the harmonic mean of precision and recall, providing a single measure of quality for the classifier when the balance between precision and recall is important. The closer this score is to 100%, the better the model's performance is in terms of both precision and recall. This high F1-score insinuates that G-GAN maintains a robust balance between the two. An F1-score of 94.52% for WGAN-GP indicates a good balance between precision and recall, though it falls short compared to G-GAN, which implies that there might be room for improvement in either precision or recall. The F1-score of 92.48% is the lowest among the three models, suggesting that BiGAN might need to improve either its precision or its recall.

TABLE II. COMPARISON OF FINAL PERFORMANCE METRICS

Model	Accuracy	Recall	Precision	F1-Score
G-GAN	97.12	97.24	96.20	96.72
WGAN-GP	96.33	95.14	94.03	94.52
BiGAN	95.00	93.37	92.01	92.48

VII. LIMITATIONS AND FUTURE WORK

One of the potential limitations of this study is its reliability on the UNSW-NB15 dataset, which, while comprehensive, may not cover all possible real-world scenarios and emerging threats faced by existing network infrastructures. Additionally, the complexity of G-GAN poses challenges in computational resource requirements and training time, which, despite the efficiency improvements observed, may still present constraints for real-time application in more complex network environments. Further studies are expected to enhance the G-GAN model, particularly in customizing it to predict and counteract zero-day vulnerabilities within network slices, which is a critical aspect of cybersecurity. An important avenue for future work will be to optimize the computational efficiency of G-GAN to ensure its viability for deployment in real-time security systems. This involves not only hardware and software advancements, but also algorithmic innovations that could reduce the complexity of the model without compromising its performance. In addition to computational optimizations, expanding the diversity and realism of datasets used for training and testing purposes could substantially enhance the robustness and generalizability of the G-GAN model. By incorporating a broader spectrum of network behaviors and attack vectors, the G-GAN framework can be further enhanced to meet the evolving requirements of network security in an ever-changing digital landscape.

VIII. SUMMARY

The development of the G-GAN model is a significant advancement in enhancing security for dynamic network slicing, a pivotal component of the 5G infrastructure. The findings show that the G-GAN model outperforms traditional GAN frameworks, such as BiGAN and WGAN-GP, in key performance indicators and also showcases a marked improvement in learning adaptive behaviors related to network security. The superiority of the G-GAN model is reflected in its ability to accurately model and interpret complex relationships within network traffic data. Through this, the G-GAN model demonstrates enhanced precision in anomaly detection, exhibiting a precision score of 96.20%, closely followed by a recall at 97.24%, and converging in an F1-Score of 96.72%. These figures notably surpass those of WGAN-GP and BiGAN, stressing G-GAN's improved functionality in differentiating between legitimate and malicious network behaviors. Moreover, the integration of G-GAN within dynamic network slicing addresses the essential requirement for a security framework that is both adaptable and quick to respond. This aligns with the growing demand for cybersecurity mechanisms that can keep pace with the rapid evolutions observed in network configurations and threat landscapes.

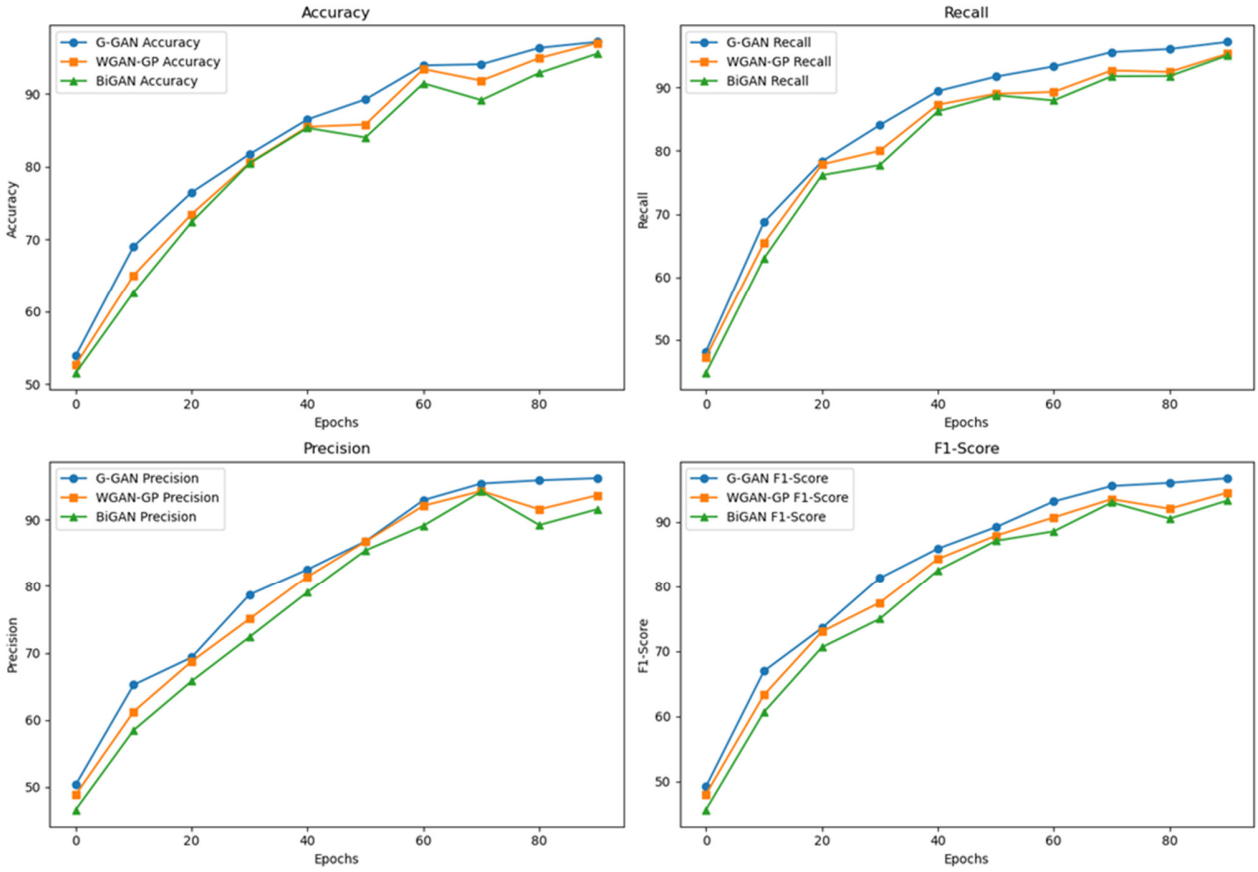

Fig. 11. Comparison of G-GAN, WGAN-GP, and BiGAN.

IX. CONCLUSION

The enhanced performance and adaptability of the proposed G-GAN model make it a promising candidate to safeguard the dynamic, heterogeneous architectures characteristic of the modern 5G networks. The proposed G-GAN model has demonstrated superior performance over baseline models in the domain of dynamic network slicing security. The proposed model exhibited enhanced precision in identifying and mitigating security threats, a testament to its robust adaptive learning capabilities and high-level understanding of network behaviors. The results suggest a move toward more intelligent and anticipatory security systems capable of safeguarding complex digital ecosystems against increasingly sophisticated threats. G-GAN's intelligent design closely aligns with the needs of modern, dynamic networks, providing a leap forward in safeguarding the integrity and functionality of network slices. This study paves the way for the integration of G-GAN models into practical security applications within 5G networks, urging the community to consider adaptive and intelligent security solutions as a new standard.

ACKNOWLEDGEMENT

The authors extend their appreciation to the Deanship of Scientific Research at Northern Border University, Arar, KSA for funding this research work through the project number 'NBU-FFR-2024-1180-01'.

REFERENCES

[1] A. A. Barakabitze, A. Ahmad, R. Mijumbi, and A. Hines, "5G network slicing using SDN and NFV: A survey of taxonomy, architectures and future challenges," *Computer Networks*, vol. 167, Feb. 2020, Art. no. 106984, https://doi.org/10.1016/j.comnet.2019.106984.

[2] S. D'Oro, F. Restuccia, A. Talamonti, and T. Melodia, "The Slice Is Served: Enforcing Radio Access Network Slicing in Virtualized 5G Systems," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, Paris, France, Apr. 2019, pp. 442–450, https://doi.org/10.1109/INFOCOM.2019.8737481.

[3] A. Kaloxylos, "A Survey and an Analysis of Network Slicing in 5G Networks," *IEEE Communications Standards Magazine*, vol. 2, no. 1, pp. 60–65, Mar. 2018, https://doi.org/10.1109/MCOMSTD.2018.1700072.

[4] F. Salahdine, Q. Liu, and T. Han, "Towards Secure and Intelligent Network Slicing for 5G Networks," *IEEE Open Journal of the Computer Society*, vol. 3, pp. 23–38, 2022, https://doi.org/10.1109/OJCS.2022.3161933.

[5] "Minimum requirements related to technical performance for IMT-2020 radio interface(s)," International Telecommunication Network, ITU-R M.2410-0, Nov. 2017.

[6] "NGMN 5G White Paper," Next Generation Mobile Networks Ltd, Feb. 2015.

[7] "Ericsson Mobility Report," Ericsson, Jun. 2020.

[8] R. Khan, P. Kumar, D. N. K. Jayakody, and M. Liyanage, "A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 196–248, 2020, https://doi.org/10.1109/COMST.2019.2933899.

[9] M. A. Ferrag, L. Maglaras, A. Argyriou, D. Kosmanos, and H. Janicke, "Security for 4G and 5G cellular networks: A survey of existing authentication and privacy-preserving schemes," *Journal of Network and Computer Applications*, vol. 101, pp. 55–82, Jan. 2018, https://doi.org/10.1016/j.jnca.2017.10.017.

[10] I. Ahmad, T. Kumar, M. Liyanage, J. Okwuibe, M. Ylianttila, and A. Gurtov, "5G security: Analysis of threats and solutions," in *2017 IEEE Conference on Standards for Communications and Networking (CSCN)*, Helsinki, Finland, Sep. 2017, pp. 193–199, https://doi.org/10.1109/CSCN.2017.8088621.

[11] A. A. Mir, M. F. Zuhairi, and S. Musa, "Graph Anomaly Detection with Graph Convolutional Networks," *International Journal of Advanced Computer Science and Applications (IJACSA)*, vol. 14, no. 11, 2023, https://doi.org/10.14569/IJACSA.2023.0141162.

[12] R. F. Olimid and G. Nencioni, "5G Network Slicing: A Security Overview," *IEEE Access*, vol. 8, pp. 99999–100009, 2020, https://doi.org/10.1109/ACCESS.2020.2997702.

[13] M. K. Hassan, S. H. Sayed Ariffin, S. K. Syed-Yusof, N. E. Ghazali, and K. A. Obeng, "A Short Review on the Dynamic Slice Management in Software-Defined Network Virtualization," *Engineering, Technology & Applied Science Research*, vol. 13, no. 6, pp. 12074–12079, Dec. 2023, https://doi.org/10.48084/etasr.6394.

[14] M. K. Hassan *et al.*, "DLVisor: Dynamic Learning Hypervisor for Software Defined Network," *IEEE Access*, vol. 11, pp. 84144–84167, 2023, https://doi.org/10.1109/ACCESS.2023.3302266.

[15] G. Gardikis *et al.*, "SHIELD: A novel NFV-based cybersecurity framework," in *2017 IEEE Conference on Network Softwarization (NetSoft)*, Bologna, Italy, Jul. 2017, https://doi.org/10.1109/NETSOFT.2017.8004228.

[16] J. Lam and R. Abbas, "Machine Learning based Anomaly Detection for 5G Networks." arXiv, Mar. 06, 2020, https://doi.org/10.48550/arXiv.2003.03474.

[17] M. Mwita, J. Mbelwa, J. Agbinya, and A. E. Sam, "The Effect of Hyperparameter Optimization on the Estimation of Performance Metrics in Network Traffic Prediction using the Gradient Boosting Machine Model," *Engineering, Technology & Applied Science Research*, vol. 13, no. 3, pp. 10714–10720, Jun. 2023, https://doi.org/10.48084/etasr.5548.

[18] H. N. Dai, Z. Zheng, and Y. Zhang, "Blockchain for Internet of Things: A Survey," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8076–8094, Jul. 2019, https://doi.org/10.1109/JIOT.2019.2920987.

[19] A. Nassar and Y. Yilmaz, "Deep Reinforcement Learning for Adaptive Network Slicing in 5G for Intelligent Vehicular Systems and Smart Cities," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 222–235, Jan. 2022, https://doi.org/10.1109/JIOT.2021.3091674.

[20] K. Abbas, M. Afaq, T. A. Khan, A. Mehmood, and W.-C. Song, "IBNSlicing: Intent-Based Network Slicing Framework for 5G Networks using Deep Learning," in *2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, Daegu, Korea (South), Sep. 2020, pp. 19–24, https://doi.org/10.23919/APNOMS50412.2020.9237008.

[21] R. K. Gupta, S. Mahajan, and R. Misra, "Resource orchestration in network slicing using GAN-based distributional deep Q-network for industrial applications," *The Journal of Supercomputing*, vol. 79, no. 5, pp. 5109–5138, Mar. 2023, https://doi.org/10.1007/s11227-022-04867-9.

[22] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A Survey on Emerging SDN and NFV Security Mechanisms for IoT Systems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 812–837, 2019, https://doi.org/10.1109/COMST.2018.2862350.

[23] C. Zhang, S. Yu, Z. Tian, and J. J. Q. Yu, "Generative Adversarial Networks: A Survey on Attack and Defense Perspective," *ACM Computing Surveys*, vol. 56, no. 4, Aug. 2023, Art. no. 91, https://doi.org/10.1145/3615336.

[24] I. Goodfellow et al., "Generative Adversarial Nets," in Advances in Neural Information Processing Systems, 2014, vol. 27.

[25] A. Creswell, T. White, V. Dumoulin, K. Arulkumaran, B. Sengupta, and A. A. Bharath, "Generative Adversarial Networks: An Overview," *IEEE Signal Processing Magazine*, vol. 35, no. 1, pp. 53–65, Jan. 2018, https://doi.org/10.1109/MSP.2017.2765202.

[26] H. Zenati, C. S. Foo, B. Lecouat, G. Manek, and V. R. Chandrasekhar, "Efficient GAN-Based Anomaly Detection." arXiv, May 01, 2019, https://doi.org/10.48550/arXiv.1802.06222.

[27] Z. Wang, Q. She, and T. E. Ward, "Generative Adversarial Networks in Computer Vision: A Survey and Taxonomy," *ACM Computing Surveys*, vol. 54, no. 2, Oct. 2021, Art. no. 37, https://doi.org/10.1145/3439723.

[28] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The Graph Neural Network Model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, Jan. 2009, https://doi.org/10.1109/TNN.2008.2005605.

[29] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive Representation Learning on Large Graphs," in *Advances in Neural Information Processing Systems*, 2017, vol. 30.

[30] W. Jiang, J. Luo, M. He, and W. Gu, "Graph Neural Network for Traffic Forecasting: The Research Progress," *ISPRS International Journal of Geo-Information*, vol. 12, no. 3, Mar. 2023, Art. no. 100, https://doi.org/10.3390/ijgi12030100.

[31] M. K. Hassan *et al.*, "Dynamic Learning Framework for Smooth-Aided Machine-Learning-Based Backbone Traffic Forecasts," *Sensors*, vol. 22, no. 9, Jan. 2022, Art. no. 3592, https://doi.org/10.3390/s22093592.

[32] E. Caville, W. W. Lo, S. Layeghy, and M. Portmann, "Anomal-E: A self-supervised network intrusion detection system based on graph neural networks," *Knowledge-Based Systems*, vol. 258, Dec. 2022, Art. no. 110030, https://doi.org/10.1016/j.knosys.2022.110030.

[33] J. Donahue, P. Krähenbühl, and T. Darrell, "Adversarial Feature Learning." arXiv, Apr. 03, 2017, https://doi.org/10.48550/arXiv.1605.09782.

[34] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved Training of Wasserstein GANs," in *Advances in Neural Information Processing Systems*, 2017, vol. 30.

[35] N. Moustafa and J. Slay, "UNSW-NB15: a comprehensive data set for network intrusion detection systems (UNSW-NB15 network data set)," in *2015 Military Communications and Information Systems Conference (MilCIS)*, Canberra, Australia, Nov. 2015, https://doi.org/10.1109/MilCIS.2015.7348942.