Received XX Month, XXXX; revised XX Month, XXXX; accepted XX Month, XXXX; Date of publication XX Month, XXXX; date of current version XX Month, XXXX.

Digital Object Identifier 10.1109/OJCOMS.2022.1234567

Towards a Fully-Observable Markov Decision Process with Generative Models for Integrated 6G-Non-Terrestrial Networks

A. Machumilane*§**, P. Cassara**§**, A. Gotta**§

[1]Department of Information Engineering, University of Pisa
[2]Institute of Information Science and Technologies (ISTI), CNR, Pisa

CORRESPONDING AUTHOR: A. Machumilane (e-mail: achilles.machumilane@phd.unipi.it).

This work was supported by the European Union under the Italian National Recovery and Resilience Plan (PNRR) PE00000001 - program ”RESTART”, the European Union under the Italian National Recovery and Resilience Plan (PNRR) Mission 4 Issue 2 Investment 1.4 “Potenziamento strutture di ricerca e creazione di ”campioni nazionali di R&S” CN00000023 – “Sustainable Mobility Center (CNMS)” and by the HORIZON-CL4-2021-SPACE-01 project ”5G+ evoluTion to mutioRbitAl multibaNd neTwORks” (TRANTOR) No. 101081983

ABSTRACT The upcoming sixth generation (6G) mobile networks require integration between terrestrial mobile networks and non-terrestrial networks (NTN) such as satellites and high altitude platforms (HAPs) to ensure wide and ubiquitous coverage, high connection density, reliable communications and high data rates. The main challenge in this integration is the requirement for line-of-sight (LOS) communication between the user equipment (UE) and the satellite. In this paper, we propose a framework based on actor-critic reinforcement learning and generative models for LOS estimation and traffic scheduling on multiple links connecting a user equipment to multiple satellites in 6G-NTN integrated networks. The agent learns to estimate the LOS probabilities of the available channels and schedules traffic on appropriate links to minimise end-to-end losses with minimal bandwidth. The learning process is modelled as a partially observable Markov decision process (POMDP), since the agent can only observe the state of the channels it has just accessed. As a result, the learning agent requires a longer convergence time compared to the satellite visibility period at a given satellite elevation angle. To counteract this slow convergence, we use generative models to transform a POMDP into a fully observable Markov decision process (FOMDP). We use generative adversarial networks (GANs) and variational autoencoders (VAEs) to generate synthetic channel states of the channels that are not selected by the agent during the learning process, allowing the agent to have complete knowledge of all channels, including those that are not accessed, thus speeding up the learning process. The simulation results show that our framework enables the agent to converge in a short time and transmit with an optimal policy for most of the satellite visibility period, which significantly reduces end-to-end losses and saves bandwidth. We also show that it is possible to train generative models in real time without requiring prior knowledge of the channel models and without slowing down the learning process or affecting the accuracy of the models.

INDEX TERMS NTN, Satellite, Generative Models (GMs), Reinforcement Learning, Actor-Critic, Multipath, Traffic Splitting.

I. INTRODUCTION

THE sixth generation (6G) mobile communications system, also known as *International Mobile Telecommunications (IMT) for 2030 and beyond* (IMT-2030) [1], will not only be an evolution of cellular networks, but a complete revolution of current terrestrial mobile networks and the end-user communication experience. According to the Working Party 5D (WP5D) [1], a group of the International Telecommunication Union (ITU) Radiocommunication Sector for IMT systems, 6G is expected to provide seamless connectivity not only to users but also to massive machine-type devices. Three main scenarios for 6G have been identified. The first scenario is Immersive Communication, an evolution of 5G enhanced Mobile BroadBand (eMBB) but with new use cases such as extended reality (XR) and holographic communication which require more bandwidth than 5G eMBB. The second scenario is Massive Communication, which assumes 5G Massive Machine Type Communication (mMTC) but aims to increase connection density, i.e. connecting many devices in a small area, using technologies such as Internet of Things (IoT), Internet of Everything (IoE) and Industrial IoT (IIoT). The third scenario is hyper-reliable and low-latency communications, which will evolve 5G Ultra-Reliable and Low Latency Communications (URLLC) to support use cases such as remote telesurgery, fully autonomous driving, industrial control and operations. In general, 6G is expected to address the shortcomings of current mobile networks and respond to growing communications needs by offering ultra-high peak data rates of around 200 Gbit/s compared to 20 Gbit/s in 5G, ultra-low latency, wide coverage and high connection density, Quality of Service (QoS) and energy efficiency, high sensing resolution and accuracy, and high security and privacy [2]. Two other important advances in 6G are the incorporation of ubiquitous and distributed Artificial Intelligence (AI) at all levels of communication [3] and the paradigm shift from network-centric to user-centric communication, where users can collaborate with the network to decide on the service they expect from the network and the allocation of channel resources.

Despite the rapid evolution of terrestrial mobile networks, supporting the 6G communications requirements described above requires new and advanced communications technologies, infrastructures, and standards. The WP5D has called for urgent research and innovation in the design of future network infrastructures and the development of various enabling technologies to support new 6G scenarios and use cases. Several enabling technologies for 6G have been identified, including the application of data and AI in distributed and collaborative ways, Integrated Sensing and Communications (ISAC), Reconfigurable Intelligent Surface (RIS), Full Duplex Operation, Radio Access Network (RAN) Slicing and Infrastructure Sharing, among others [1]. In addressing the 6G requirement for wide coverage and full connectivity, the ITU report on Future Technology Trends for Terrestrial International Mobile Telecommunications systems towards 2030 and Beyond [1] applied to 6G what the Third Generation Paternship Project (3GPP) proposed for 5G [4] and recommends integrating 6G mobile networks with Non-Terrestrial Network (NTN) technologies. NTN platforms are network segments that use transmission equipment or base stations mounted on an airborne or spaceborne vehicle. NTN platforms include satellites such as geosynchronous (GEO), Medium Earth Orbit (MEO) and Low Earth Orbit (LEO), High Altitude Platforms (HAPs), and Unmanned Aerial Systems (UASs). The white paper on 6G wireless networks [5] also recommends that future wireless networks must be able to connect seamlessly with terrestrial and satellite networks. Since satellites have wide coverage, they can complement terrestrial mobile networks in partially connected and unconnected areas such as maritime areas, mountainous regions, and deserts. Although satellites have not been widely used in the past due to high construction costs, as technology advances and communication requirements increase, various satellite constellations such as Starlink, OneWeb, and Telesat [6] have been launched. High Altitude Platform (HAP) systems include airborne base stations deployed above 20 km and below 50 km to provide wireless access to devices in large areas. HAP systems can be used as HAP Stations (HAPS) to offer internet access between fixed points in suburban and rural areas and in emergency situations [7]. HAPS offer wide coverage, flexible deployment, and low construction costs. They also have low latency due to their relatively lower altitude compared to satellites. Another application of HAP systems is to use HAPS as International Mobile Telecommunication (IMT) Base Station (HIBS) to complement IMT requirements for mobile phones or other terminals in areas not covered by HAPS. So, with HIBS, some of the access functionalities in the terrestrial networks can be moved to the non-terrestrial infrastructure. UASs, commonly known as Unmanned Aerial Vehicles (UAVs) or drones, can also be used as IMT base stations. UAVs have attracted a lot of attention because they are lightweight, easy to deploy, and offer flexible services. Exploiting the advantages of terrestrial networks and non-terrestrial platforms will support a range of new applications and use cases such as remote monitoring, rescue operations, reconnaissance, goods delivery, connected autonomous vehicles (CAVs), and high-speed transportation (e.g., trains or aircraft). In this paper, we focus on the integration between LEO satellites and the upcoming 6G mobile networks.

The main challenge in integrating terrestrial IMT and NTN is the channel modeling of the service link, i.e. the link between the NTN terminal or User Equipment (UE) and the satellite or an NTN platform, as this link requires Line-of-Sight (LOS), which is impaired when both the satellite and the UE are in relative motion. In dense urban scenarios, tall buildings, and other tall infrastructure can severely degrade LOS communications as signals are blocked or reflected. In addition, the LOS probability varies with the elevation angle of the satellite, with low elevation angles having a low LOS probability due to blocking. The LOS variations can lead to unreliable communication due to poor connectivity, network unavailability, or service interruption, making it difficult to meet 6G communication requirements. Existing ITU service link models take into account the elevation angle, frequency, and propagation environment (e.g. urban or rural), [8] but not the relative movement of the UE and satellite, which can make the propagation environment non-stationary because the LOS probability may vary with time.

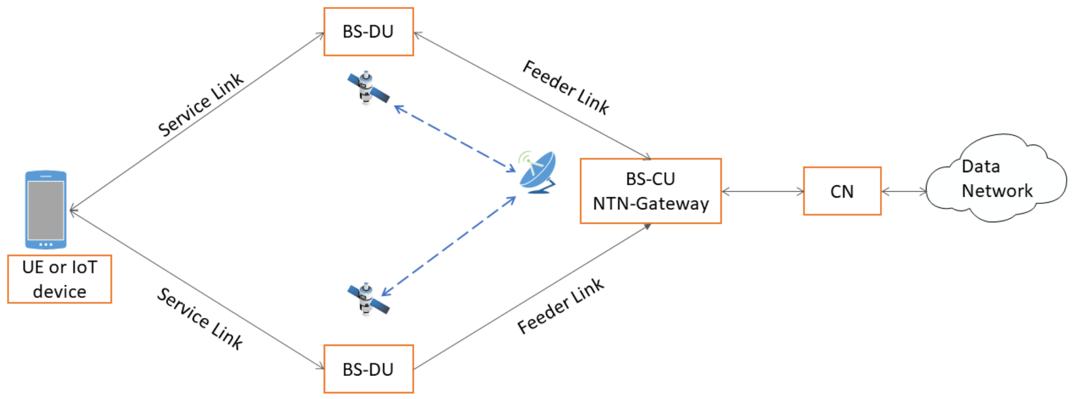


FIGURE 1: Reference Scenario: A user equipment in a duo-connectivity in Terrestrial-NTN integrated network. The UE accesses the IMT core network and the Internet via two LEO satellites with RAN capabilities.

II. Reference Scenario and Motivations

In this paper, we propose an AI-based intelligent system for LOS estimation and traffic scheduling on the access link of 6G-NTN integrated networks. We use the Actor-Critic (AC)-Reinforcement Learning (RL) framework, in which an RL agent continuously monitors and learns the LOS probability of multiple links and selects an appropriate subset of the available links on which to schedule traffic to increase link availability and reliability by increasing the probability of good traffic reception. Since our proposed framework is not deterministic but learning-based, it can track the dynamic variations of LOS due to terrain and mobility. As shown in Figure 1, our reference scenario, the UE with multiple interfaces can connect to two satellites in multi-connectivity mode. The two satellites are equipped with BS through which the UE connects to the terrestrial IMT Core Network (CN) and theData Network (DN). The UE can be any user terminal, a UAV, or an IoT device. Our RL agent learns the channel characteristics of each access link and schedules traffic according to link characteristics such as LOS and Packet Loss Rate (PLR) to increase link availability, reliability and throughput. Given the limited computational resources of the UE, the RL agent can be deployed on the edge device with high computational resources or anywhere in the network and offered as AI-as-a-Service (AIaaS) as envisaged in 6G [1]. To further improve link reliability and throughput, we use a multipath transmission technique that splits a single traffic flow into sub-flows and transmits each subflow over a separate path, achieved by one or more communication channels, to increase the probability of good reception by leveraging the different link characteristics. We then couple multipath with traffic duplication, which adds redundancy to further increase the probability of good reception because the redundancy traffic is transmitted on different links than the information traffic, so that traffic lost on one link can be recovered on other links. We perform redundancy optimisation to avoid excessive bandwidth consumption.

To support duo-connectivity and multipath transmission, we use the standard mechanism known as Access Traffic Steering, Switching, and Splitting (ATSSS), originally introduced by 3GPP for IMT-2020 [9], but needs to be further developed and improved for IMT-2030 to support Multi-Access Packet Data Unit (MA-PDU) session services through self-learning decision policies supported by AI. Access traffic steering means the selection of an access network over which a particular new data flow is to be transmitted. On the other hand, traffic switching refers to the process of moving all the traffic of an ongoing flow from one access network to another while maintaining the continuity of the flow. On the other hand, traffic splitting refers to the process of dividing a data flow into parts that are transmitted over different access networks. 3GPP standard defines two ATSSS functionalities: ATSSS high-layer functionality and ATSSS low-layer functionality (ATSSS-LL). In the former, traffic steering is performed above the Internet Protocol (IP) layer, where each substream is identified with a unique IP address, as shown in Figure 2. Link monitoring and performance measurements such as PLR or Round-Trip Time (RTT) are performed End-to-End (E2E) between the UE and the DN through a multipath server proxy in the core and can be used as criteria for traffic steering decisions. The standard identifies two protocols for ATSSS higher-layer functionality: Multi-Path-TCP (MPTCP) for multipath Transmission Control Protocol (TCP) traffic and Multi-Path-QUIC (MPQUIC) for Quick UDP Internet Connections (QUIC) User Datagram Protocol (UDP) traffic. ATSSS-LL, on the other hand, is implemented at the link layer, where Media Access Control (MAC) addresses identify subflows and can handle any traffic, including TCP, UDP and Ethernet traffic. ATSSS is a very important feature for the 6G paradigm shift from a network-centric to a user-centric approach, as it supports collaborative network performance measurements between the network and the user. The user can measure access link performance in terms of LOS, delay, PLR, bandwidth, link availability, or unavailability and either share the measurements with the core network or use the measurements autonomously for uplink (UL) traffic steering over the access networks. In the future, this feature can be used to support UE decisions on channel resource allocation, which is one of the provisions in the user-centric 6G networks.

The framework proposed in this paper performs traffic splitting and steering at the link layer in accordance with the ATSSS-LL functionality. The ATSSS standard has introduced a function called Performance Measurement Function (PMF) that enables the exchange of messages between the UE and the core for performance measurements. We have developed a stub that provides our learning agent with link performance measurements such as LOS, link-PLR, and E2E-PLR for traffic steering over the two satellite networks. LOS and link-PLR are used to decide which link to steer the traffic to, while E2E loss is used to decide whether to use a single transmission or multiple transmissions with traffic duplication to compensate for E2E losses. In this case, the E2E loss occurs when no traffic is received on either link. Since the LOS changes with the elevation angle of the satellite, the RL agent constantly retrains to track the ever-changing LOS of multiple moving satellites and allows the UE to distribute traffic to the appropriate link(s). However, we found that each time the elevation angle changes, the agent takes a long time to re-train and converge compared to the duration of the satellite visibility [10]. Normally, a moving satellite is visible from a UE near Earth or on Earth for a certain period of time called the satellite visibility period, which can be very short for large constellations. For example, the satellite visibility period in Paris, France, was found to be 3.5 minutes for Starlink constellations. During this visibility period, the satellite changes its elevation angle and consequently, the LOS probability also changes. If the learning agent converges slowly, it cannot make the best use of the satellite visibility period because the elevation angle and LOS probability change before it converges. As a result, the agent transmits with non-optimal policies.

In this paper, we use generative models to solve the problem of slow convergence of the learning agent. Although there can be several reasons for slow convergence, we focus our investigation on learning-based LOS estimation, which in such scenarios is modeled as Partially Observable Markov Decision Process (POMDP) [11], since the learning agent can only observe the states of the links it selects for transmission at a given time, for scalability reasons[1]. With multiple channels, the agent needs a lot of time to fully know the states of all available channels and to select the appropriate channels. The obvious and simple solution would be to duplicate the traffic and transmit it over all available links to quickly learn the LOS probability of each link. Although this seems to be a simple solution, it is inefficient as it wastes bandwidth. In this work, we provide a more efficient and intelligent solution that transforms the POMDP into a Fully Observable Markov Decision Process (FOMDP) so that the agent can have the CSI of all available channels, including the channels it does not select on each transmission event, without having to transmit on all links. To this end, we use deep generative models (Generative Models (GMs)) which we train to generate synthetic channel states that closely resemble the real channel state of the links not selected by the agent. Specifically, we use two deep GMs [12]: Conditional Tabular Generative Adversarial Networks (CTGANs), a version of the most popular and powerful deep generative model called Generative Adversarial Network (GAN), and Tabular Variational Autoencoders (TVAEs), a variant of Variational Autoencoder (VAE), another powerful and commonly used deep generative model. When the agent selects a subset of the available channels at each transmission event and learns their LOS probability, the trained GMs generates synthetic LOS estimates for the remaining subset. In this way, the agent has a complete view of the channel states for each transmission event. As a result, the agent learns quickly, converges faster, and transmits with an optimal policy for most of the satellite visibility period. As explained in Section VI, the GMs can be trained offline or during deployment.

Our main contributions can be summarized as follows:

1) We propose the use of reinforcement learning (RL) and generative models (GMs), to provide intelligence into integrating terrestrial and non-terrestrial networks for supporting 6G communication requirements such as improved network accessibility and connectivity, link availability and reliability, and high data rates.
2) We use generative models, specifically GANs and VAEs, to transform a POMDP into a FOMDP. The GMs generate synthetic states of a partially observable Markov process that are not visited by the agent during the learning process and, thus, transform a partially observable process into a fully observable Markov decision process by providing the agent with a complete view of all states. This method can be applied not only to LOS estimation, as in this work, but also to any partially observable Markov decision process. To the best of our knowledge, this is the first work that uses generative models to transform a POMDP into a FOMDP.
3) We develop an actor-critic-RL framework to estimate the LOS probability of multiple service links between UE and LEO satellites in IMT-NTN integrated networks with heterogeneous characteristics. The RL agent learns to determine the LOS probability of each link and select an appropriate subset of the available links for transmission, i.e., the link(s) with a relatively higher LOS probability, to increase the probability of good traffic

[1]Channel State Information (CSI) analytics have computational and storage costs, and if multiple interfaces can be used, a policy to limit data collection must be considered. Therefore, limiting the analysis to only one interface being used at any given time can be a reasonable choice.

reception, improve link availability and reliability, and increase data rates.

4) We couple multipath with traffic duplication to proactively compensate for E2E losses and consequently increase throughput. Since traffic duplication can increase bandwidth consumption, we optimize the use of redundancy to avoid excessive bandwidth consumption. We show through intensive simulations that our RL agent can track low E2E losses when deployed in different propagation environments with different E2E loss thresholds according to the end-user QoS agreement.
5) Since the satellite visibility period is shorter than the convergence time of the RL agent, we use our proposed model for transforming a POMDP into a FOMDP, to convert a learning-based LOS estimation which is a POMDP, into a FOMDP to accelerate the convergence of the RL agent within the satellite visibility period. We use GANs and VAEs to generate synthetic LOS link states of the links not visited by the agent and thus, convert a POMDP into a FOMDP since the RL agent now has complete knowledge of the LOS state of all links. This allows the agent to learn and converge within a short time, and transmit with an optimal policy for most of the satellite visibility period.
6) Finally, we show through simulations that GMs training can be performed in real-time without slowing down the RL agent learning process or affecting GMs accuracy.

The rest of the paper is organized as follows: In Section III, we review the state-of-the-art techniques with respect to our work. We present our system model in Section IV and describe the training and evaluation of the GMs in Section V. Section VI presents the architecture and training of the Actor-Critic Reinforcement Learning Agent while its performance evaluation is presented in Section VII. Section VIII concludes the paper and identifies future research directions.

III. Related Work

A. LOS Estimation and Traffic Scheduling

Several methods for estimating LOS and scheduling traffic through multiple channels have been suggested. In [13], a theoretical model for LOS prediction in cloud-free sky is proposed which takes into account the angle between the satellite and the ground station. In [14], a maximum likelihood-based method for detecting the presence of Non-Line-of-Sight (NLOS) is proposed. In [15], the authors propose an empirical model for probability estimation of LOS for satellite and HAPs communications. All of these approaches are empirical and deterministic and therefore not suitable for dynamic and nonstationary NTN propagation environments. Traditional and static traffic scheduling techniques such as Round-Robin (RR), Weighted Round Robin (WRR) have been shown to be inefficient in heterogeneous and time-varying wireless channels [16]. With the pursuit of self-reconfigurable networks, the improved schedulers such as deficit round robin (DRR) and weighted fair queuing (WRQ) schedulers [16], RTT, PLR, [17], the lowest-RTT-first schedulers [16], [18], [19] are becoming increasingly unpopular and research is leaning towards learning-based schedulers. For example, in [20] a Deep-Q (DQ) RL-based scheduler is presented for dynamically allocating bandwidth to different WiFi applications. Wu et al. [21] have proposed a RL-based multipath scheduler for multipath QUIC on WiFi and cellular applications. In [22], a AC agent is used for multi-channel access in wireless networks to avoid collisions. Yang et al. [23] propose an AC-based scheduler for cognitive Internet-of-Things (CIoT) systems. Another AC-based scheduler is proposed in [24] to address end-to-end delay in Fog-based IoT systems. However, all these works are partially observable processes that may suffer from the slow convergence of the learning agent. We aim to address this problem in this work by using GMs to transform a POMDP into a FOMDP. Since our proposed framework is designed for multipath systems, it provides not only a scheduling mechanism but also traffic protection. We schedule traffic by steering and splitting it over multiple paths to increase the probability of good reception leveraging the different path properties as in [25]. Our framework also avoids delays caused by traffic protection systems such as Automatic Repeat reQuest (ARQ) that uses retransmissions to compensate for the loss, which may be unsuitable for satellite communications with large propagation delays. In addition, our system limits the waste of bandwidth like some layered Forward Error Correction (FEC)-based systems do [26], which are difficult to use with fixed coding rates in dynamic contexts, and avoids introducing delays due to the encoding-decoding chain [27] as well as further complexity.

B. Deep Generative Models

Deep generative models have attracted much attention and found several applications, especially in computer vision, including the generation of realistic images, videos, music relics, texts, and language processing. In [28], [29], and [30], GANs are used for image generation, while the authors in [31] use VAE and GANs to generate videos from texts. GMs are used in [32] to improve the quality of the training dataset for Electrocardiogram (ECG) signal classification. Although the application of GMs for communication is still being explored, some work has already been proposed. For example, in [33], the authors use VAE to generate channel parameters such as path loss, delay, and arrival and departure angles. They first estimate the LOS and NLOS state of a link using a ray tracer and use these estimates to train VAE and generate other channel parameters. The use of VAEs and GANs to improve the LOS estimation was also discussed and compared in [34], with a similar scenario, while the use of a federated approach with VAEs was introduced in [35], and investigated for the first time. The Conditional GAN (cGAN) is used in [36] to model channel effects in an E2E wireless network and optimize receiver gain and decoding. In particular, the cGAN is used to support

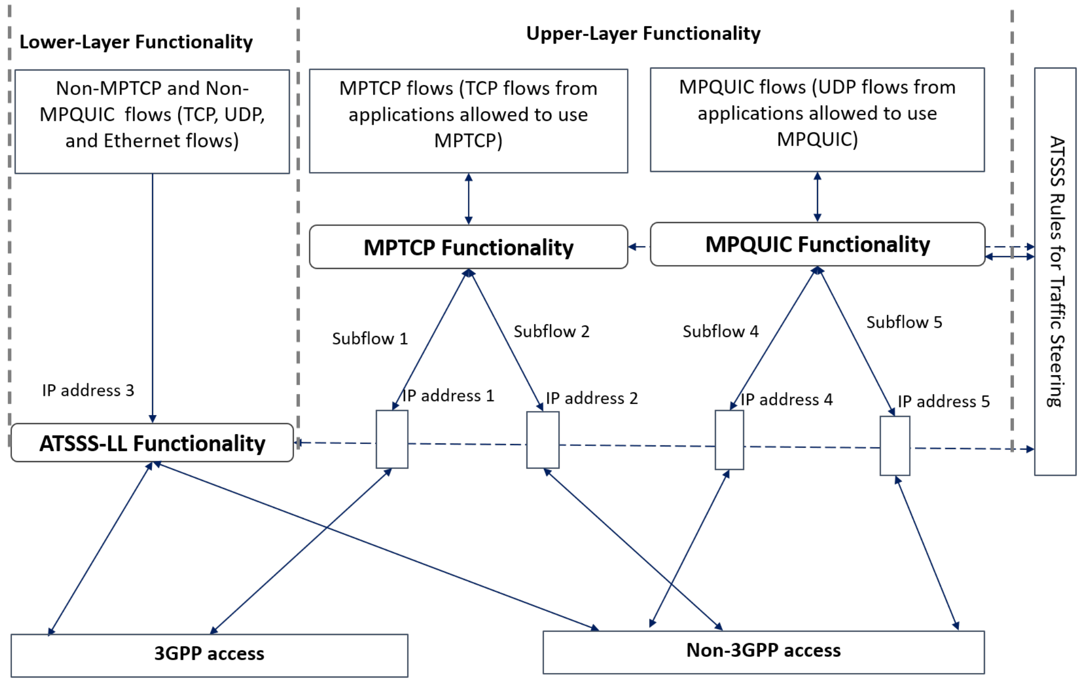

Lower-Layer Functionality
Upper-Layer Functionality
Non-MPTCP and Non-MPQUIC flows (TCP, UDP, and Ethernet flows)
MPTCP flows (TCP flows from applications allowed to use MPTCP)
MPQUIC flows (UDP flows from applications allowed to use MPQUIC)
ATSSS Rules for Traffic Steering
MPTCP Functionality
MPQUIC Functionality
Subflow 1
Subflow 2
Subflow 4
Subflow 5
IP address 3
IP address 1
IP address 2
IP address 4
IP address 5
ATSSS-LL Functionality
3GPP access
Non-3GPP access

FIGURE 2: ATSSS Functionalities in a UE model. The EU connects to the Internet via two access networks: 3GPP and non-3GPP. With upper-layer functionality (MPTCP and MPQUIC) the data flow is split above the network layer while with lower-layer functionality (ATSSS-LL) flow splitting is done at the link layer. [9]

the learning of the Deep Neural Networks (DNNs)-based communication system when the CSI is unknown. This work is similar to our study in which we use the CTGANs and TVAEs to generate missing LOS estimates for the AC-based transmission system to improve the QoS by reducing E2E losses.

IV. System Model

The WP5D group has recommended that the existing 3GPP architecture for integrating terrestrial IMT and NTN also be used for the integration of 6G mobile networks with NTN, where the Base Station (BS) is split into Distributed Unit (DU) and Centralized Unit (CU) [1]. Although the WP5D group has not specified the placement of the DU and CU, the existing 3GPP [4] provides that the DU can be mounted on the satellite, while the CU forms part of the terrestrial infrastructure. As shown in Figure 1, the two satellites have a DU on board to provide BS functionalities. The UE accesses the network via these satellites in a multi-connectivity mode and connects to the CN and the DN via a common CU on the ground. We use the StarLink LEO satellite constellations [37].

A. Channel Model

In this work, we adopt the channel model provided by ITU [38] for designing Earth-space communication systems. We simplify this model using the Lutz approach [39], [40] and assume two channel states: the good state (G) and the bad state (B). The good state is characterized by the presence of the LOS, and good traffic reception and is modeled by a Rician fading model for unshadowed areas. The bad state, on the other hand, is marked by NLOS, losses or bad reception and is modeled using the Rayleigh fading model. We adopt these models to compute the channel state transition probabilities which we use to create the dataset to train our learning agent and the generative models. For the sake of simplicity, in this work, we did not consider interference.

Computation of the Link State Transition Probabilities

We define the transition matrix as follows [27]:

$$\Psi = \begin{bmatrix} 1-P_b & P_b \\ P_g & 1-P_g \end{bmatrix}$$

where P_b is the probability to transition from good state to bad state and P_g from bad state to good state. It follows that,

$$T_g = 1/P_b; T_b = 1/P_g \tag{1}$$

where T_g and T_b indicate the time duration of the good and bad states respectively and are given as follows:

$$T_b = \frac{r}{v \cdot k} d_g; \, T_g = \frac{r}{v \cdot k} d_b, \tag{2}$$

where, d_g and d_b are the mean duration of the good and bad states [39]; v is the speed of the UE in (m/s) transmitting packets of size k bits at a rate r.

Since the LOS probability depends on the elevation angle of the satellite, the ITU recommendation [38] provides sta-

TABLE 1: Earth-Space Link parameters at 2.2 GHz in France [38]

Elevation angle	$\boldsymbol{\mu_{g,b}}$	$\boldsymbol{\sigma_{g,b}}$	$\boldsymbol{t_{ming,b}}$
20°	2.0042, 3.6890	1.2049, 0.9796	3.9889, 10.3114
30°	2.7332, 2.7582	1.1030, 1.2210	7.3174, 5.7276
45°	3.0639, 2.9108	1.6980, 1.2602	10.0, 6.0
60°	2.8135, 2.0211	1.9595, 0.6568	10.0, 1.9126
70°	4.2919, 2.1012	2.4703, 1.0341	118.3312, 4.8569

TABLE 2: Earth-Space Link LOS transition probabilities

Elevation angle	$\boldsymbol{P(Bad \rightarrow\ Good)\ (P_g)}$	$\boldsymbol{P(Goog \rightarrow\ Bad)\ (P_b)}$
20°	0.00014310	0.00047466
30°	0.00024460	0.00027570
45°	0.00020318	0.00007556
60°	0.00105161	0.00010797
70°	0.00052923	2.76683×10^{-6}

tistical parameters to determine the mean duration d_g and d_b of the good and bad states respectively at different elevation angles, frequency, and different propagation environments, such as urban and rural. In this work, we use the parameters for the urban environment at 2.2GHz as reported in Table 1. These parameters are the statistics of the duration of the good and bad states which include the mean $\mu_{G,B}$, the standard deviation $\sigma_{G,B}$, and the minimum duration d_{min} of each state. Substituting these parameters in equation (3), we calculate the mean duration d_g and d_b.

$$d_{g,b} = exp\left(\mu_{g,b} + \frac{\sigma^2_{g,b}}{2}\right) \frac{erf\left(1 - \frac{log d_{min,g,b} - (\mu_{g,b} + \sigma^2_{g,b})}{\sigma\sqrt{2}}\right)}{erf\left(1 - \frac{log d_{min,g,b} - \mu_{g,b}}{\sigma\sqrt{2}}\right)} \tag{3}$$

Finally, we combine equations (1) and (2) to obtain the transition probabilities P_b and P_g as follows.

$$P_b = (\frac{r}{v \cdot k} d_g)^{-1};\ P_g = (\frac{r}{v \cdot k} d_b)^{-1} \tag{4}$$

We report the computed transition probabilities in Table 2 and use these probabilities to create Markov states dataset with LOS/NLOS traces to train our models.

B. Deep Generative Neural Networks

Deep Generative AI refers to unsupervised and semi-supervised Machine Learning (ML) Algorithms that use Neural Networks (NNs) to learn and model the distribution of the true data and generate new synthetic data with a similar distribution to the true data. GMs are used to produce high-quality images, videos, sounds, and text that closely resemble the original data. They are also used to augment data and generate large amounts of data for training other ML algorithms, using only a small amount of real data. There are many types of deep GMs, but two are most commonly used: The GAN and the VAEs. There are many variants of these two as well. In this work, we use the CTGANs, which is a variant of the GAN, and the TVAEs, which is the variant of the VAE. They are built in the TensorFlow library and belong to the Synthetic Data Vault (SDV) package. The choice of the CTGAN and TVAE was motivated by the fact that these two models can handle tabular data and therefore allow us to train only one model that can generate synthetic data for any number of available service links since they can learn the data distribution in each column of the training dataset. For training the GMs, we considered three elevation angles: 70°, 60°, and 45° and organized the training dataset into a table of three columns with each column containing the LOS/NLOS traces of one of the elevation angles or channels. Thus, knowing the data distribution in each column, a single CTGAN or TVAE model can generate synthetic data for all the columns at once, which would otherwise require training one model for each channel. Below is a brief description of the structures and functionalities of the CTGAN and TVAE.

1) CTGAN-Conditional Tabular GAN

The Generative Adversarial Network (GAN) [28] is a type of generative neural network that has become popular due to its ability to produce high-quality synthetic data. The basic architecture of the GAN consists of two neural networks, the generator and the discriminator. The generator generates synthetic data that resembles real data, while the discriminator is a classifier that attempts to distinguish fake data from real data. The generator and discriminator are trained in an adversarial way based on a two-player game theory that aims to find a Nash equilibrium [41] in which the generator tries to fool the discriminator by generating data that looks like real data, while the discriminator tries to catch the generator by distinguishing real data from fake data generated by the generator. After training, the generator is able to generate data that is too real for the discriminator to distinguish from real data. The discriminator is trained to maximize the equation shown in (5)

$$\log \mathcal{D}(x) + \log(1 - \mathcal{D}(\mathcal{G}(z))) \tag{5}$$

while the generator minimizes the equation shown in (6),

$$\log(1 - \mathcal{D}(\mathcal{G}(z))) \tag{6}$$

in both previous equations $\mathcal{D}$ and $\mathcal{G}$ are functions of the generator and discriminator networks, respectively, and z and x are noise and real data samples, respectively.

The Conditional Tabular GAN (CTGAN) is a type of GAN developed by [12] for dealing with tabular data. The original GANs were developed primarily for images and could not handle tabular data. The CTGAN is conditional in that, unlike the general GANs, it can produce data with a particular property or distribution. For example, the basic or vanilla GANs trained to generate human faces can only generate random faces as found in the training data. It cannot generate a specific face. To condition the model to generate data with specific features, patterns, or distribution, the generator, and discriminator are given additional information about the data as input. This may be labels of the training data or a particular distribution. This allows the generator to produce data with a desired distribution or property.

2) TVAE - Tabular Variational Auto-Encoders

Variational autoencoders are among the widely used unsupervised deep GMs. Like autoencoders, VAEs have a two-network structure, the encoder and the decoder. However, unlike autoencoders, VAEs are used to generate new data. The encoder maps the input real data into a compressed latent vector and the decoder generates new data from the latent vector. The VAEs differ from autoencoders because in VAEs the latent vector is regularized for generating new data. Instead of encoding an input into a single point, it is encoded as a distribution, which is then regularized by parameterization, using a normal distribution such as a Gaussian distribution so that the decoder can use any sample from it to generate new data. Equation (7) gives the loss function used to train the VAE [42].

$$\mathcal{L}(\theta_d, \theta_e) = -\mathbb{E}_{q_{\theta_e}(z|x_i)}\left[\log p_{\theta_d}(x_i|z)\right] + KL(q_{\theta_e}(z|x_i)\|p(z)) \quad (7)$$

The VAE is trained to minimize the reconstruction error (the first term of the expression) between the input data and the generated data, and to maximize the likelihood of the parameters of the Gaussian distribution (the second term of the expression) that defines the latent space. The second term acts as a regularizer to measure the loss when $q_{\theta_e}(z|x_i)$ is used to represent the distribution $p(z)$ of the latent space z. $q_{\theta_e}(z|x)$ is the distribution of the input variables x and $p_{\theta_d}(x|z)$ represents the distribution of the decoded variables, while θ_e and θ_d are the parameters of the encoder and decoder, respectively. This paper adopts the TVAEs a version of VAE available in the same package as CTGAN for handling tabular data as described above.

V. Training the CTGAN and TVAE Models

The generative models were trained in two ways. We first trained the models offline using the training data generated according to the transition probabilities in Table 2. Then, we simulated the real-time training, i.e., training the GMs when the RL is in operation. In this case, the training data is acquired by the RL as it learns the channel states. In the following, we describe the two training methods in detail and evaluate the accuracy of the GMs in each case. The training parameters are shown in Table 3. Two performance evaluation metrics were used to evaluate the performance of the trained GMs: the Kolmogorov-Smirnov Test (KS-test) and the Kullback-Leibler divergence (KL -divergence).

TABLE 3: Simulation Parameters.

Name	Value
Size of the training dataset	3000k traces
Number of epochs	100
Batch size	50
Generator and discriminator dimensions	256,256
Generator and discriminator learning rates	2e-4
TVAE encoder and decoder dimensions	128,128
Number of Transmission links (channels)	2
Number of hidden layers	3
Learning rate for the critic, α	0.03
Number of neurons for hidden layer	64
Discount Factor, γ	0.96
Learning rate for the actor, β	0.01
Optimizer	ADAM
E2E loss threshold (ξ)	$0.0001, 0.0005, 0.001, 0.01$
Total number of iterations L	1000k
Number of iterations in an episode M	1000

The KS-test measures the distance between two empirical Cumulative Density Function (CDF) and is usually presented as a complementary measure, i.e., 1 - the difference in CDF. Thus, the higher the KS-test value, the more similar the two CDFs are. In our case, we compare the CDFs of the real and synthetic data. The KL divergence, on the other hand, measures the difference between two probability distributions. The lower the KL divergence, the greater the similarity between the two distributions.

A. Training Dataset

The datasets to train the GMs and the AC-RL agent were created as follows: we used the transition probabilities computed in Section III and reported in Table 2 to create the Markov States for the LOS and NLOS for different elevation angles. The LOS was coded as 1 and NLOS as -1. Thus, the dataset consisted of a set of traces [-1,1...] for each elevation angle according to the state transition probabilities. The datasets created in this way were used to train the AC-agent in a partially observable Markov process, and the generative models in offline mode, while the dataset for real-time training of GMs consisted of the channel states collected during the learning process of the agent. The dataset to train the AC-agent in a FOMDP is a combination of the traces obtained by using the state transition probabilities (for the channels selected by the agent) and the synthetic states generated by the trained GMs (for the channels not selected by the learning agent. See Algorithms 1 and 2). The training datasets have different sizes depending on the model to be trained as described in the appropriate sections below.

B. Offline Training of generative models

The offline training involved two ways: using *separate dataset* and *combined dataset*. In the separate dataset, we used the transition probabilities given in Table 2 to create LOS/NLOS traces for each of the three channels or elevation

TABLE 4: CTGAN and TVAE accuracy and training time on separate training datasets

Elevation angle	KS-Test		KL-Divergence		Training Time (s)	
	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE
70°	0.95696	0.9541	0.0478	0.0624		
60°	0.9828	0.9634	0.0023	0.0119	5427.84	5313.05
45°	0.9831	0.9582	0.0006	0.0040		

TABLE 5: CTGAN and TVAE accuracy and training time on combined training datasets

Elevation angle	KS-Test		KL-Divergence		Training Time (s)	
	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE
70°	0.9117	0.9337	0.0405	0.0255		
60°	0.9423	0.9644	0.0156	0.0007	15925.74	15031.82
45°	0.8176	0.7956	0.107	0.1453		

angles ($70°$, $60°$, $45°$). The traces were organized in a tabular form of three columns with each channel traces for each column. The CTGANs and TVAEs models were trained to generate new traces for each column or for each channel. For the combined dataset, all the traces for the three elevation angles were combined into a one-column dataset and reshuffled to balance the data.

Generative models performance evaluation (offline training)

The accuracy of the models trained on the separate dataset was evaluated by comparing the generated traces for each channel or column with the real traces of the corresponding channel. In the case of the combined dataset, the comparison was made between the combined generated traces with the of each channel or column. Then, the two training models were compared in terms of model accuracy and training time. The aim is to find out which training mode achieves high accuracy in a short time and which model between CTGAN and TVAE performs better than the other in each training mode. Table 4 and Table 5 show the accuracy and training time for the two models trained with separate and combined datasets respectively. Accuracy is measured by the distance between the real and generated data. Figure 3 shows the comparison between the distribution (PDF) of the real and generated data for the two models trained with the separate and the combined dataset for the three channels. The results show that our models achieved very high accuracy in all scenarios, with KT -test up to 98% and KL -divergence up to 0.0006. Both models show similar performance with minor differences in all scenarios. However, the models perform better when trained on the separate dataset than on the combined dataset. This may be due to the fact that the three channels are not correlated, so combining the channels does not give good results. This means that training with a separate data set is suitable for uncorrelated channels and with a combined data set for correlated channels. In terms of training time, the results show that both models train faster with the separate dataset than with the combined dataset, with TVAEs training relatively faster than CTGANs in both cases. Based on these results, the models trained with the separate dataset were used for the remainder of this work to generate data traces for training our RL agent.

C. Real-time Training of generative models

Real-time training refers to the scenario where the GMs are trained when the RL agent is already deployed for

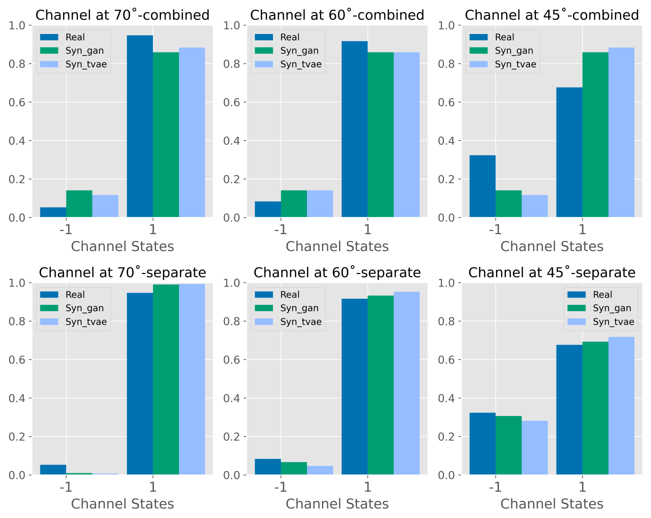


FIGURE 3: Comparison between the distributions (PDFs) of the real and synthetic data generated by CTGAN and TVAE models trained on separate and combined datasets. The datasets contain LOS/NLOS traces of satellite links at $70°$, $60°$, and $45°$.

transmission. This is a more realistic scenario that can occur when the channel model is not known in advance, which is usually the case, or when there are no LOS datasets for training the GMs. In this case, the RL agent must transmit for a certain time on all the available channels to acquire the CSI of all the channels. Then the acquired traces are used to train the GMs. Finally, the trained models are used to generate synthetic states of the channels that the agent does not select for transmission at each transmission event so that the RL agent can have a complete observation of the states of all channels. This is a very challenging scenario due to the time constraint. First, the time to acquire CSI should be very short to avoid wasting bandwidth since the agent has to transmit by duplicating traffic over all available channels. Second, the training time of the GMs should be very short because of the limited satellite visibility period. To simulate this scenario and overcome these challenges, we first created training datasets with different sizes: 2k, 5k, 10, 20k, 30k, 40k, and 50k to train the GMs. The goal is to determine the minimum size of the dataset that will train the models in the shortest possible time and achieve the highest possible model accuracy. In this way, we can evaluate whether our proposed approach is feasible for online training. We trained both the

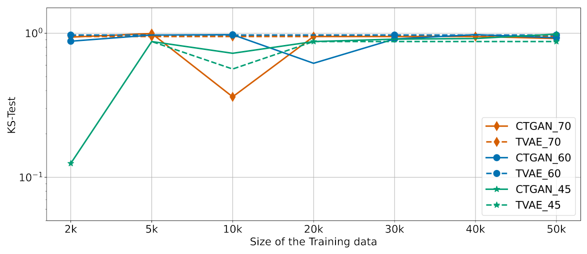


FIGURE 4: Comparison in terms of the KS-Test between the real data and synthetic data generated by CTGAN and TVAE models trained in real time with training datasets of different sizes (2k, 5k, 10k, 20k, 30k, 40k, and 50k). The datasets contain LOS/NLOS traces of the satellite links at different elevation angles (70°, 60°, and 45°)

CTGANs and TVAEs models with only a single epoch and recorded the training time for each training dataset.

Generative models performance evaluation (real-time training)

Table 6 shows the accuracy and the training time of the CTGANs and TVAEs models in terms of the KL-Divergene and the KS-Test between the real data and the synthetic data generated by the two models. The models were trained with datasets of different sizes containing the states of the satellite links at different elevation angles. The aim was to determine the minimum size of the atasets that can be used to train the models and achieve good accuracy. From these results, it can be seen that training with a 10k dataset is the best compromise, sine with this size of dataset the models train within a short time of 3.89 seconds and 2.39 seconds for CTGAN and TVAE, respectively, achieving relatively good accuracy at the three elevation angles (70°, 60°, and 45°). Figure 4 is the graphical representation of the variations of the KS-Test between the real data and synthetic data generate by CTGAN and TVAE models trained with datasets of different sizes. It can also be seen that 10k dataset achieves good accuracy for both models. The results also show that increasing the size of the dataset does not have much effect on the accuracy of the TVAE model. TVAE can thus be trained with a very small dataset and achieve a good accuracy. Figure 5 shows the variation of the training time for CTGAN and TVAE models at different training datasets. These results show that CTGAN requires longer time to train than TVAE at all the sizes of the datasets considered. In the rest of this work, we used the models trained with the 10k dataset to generate synthetic datasets to evaluate the performance of the RL agent with real-time trained GMs.

VI. Actor-Critic Reinforcement Learning

After discussing the structure, training, and evaluation of the CTGAN and TVAE models in the previous sections, in this section, we present the architecture and the learning process of our proposed Actor-Critic Reinforcement Learning framework.

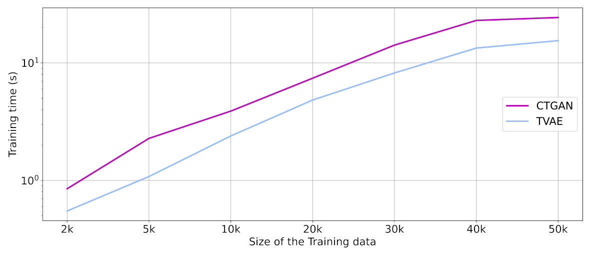


FIGURE 5: Comparison in terms of training time between CTGAN and TVAE models trained in real time with training datasets of different sizes (2k, 5k, 10k, 20k, 30k, 40k, and 50k)

A. Problem Formulation

We formulate the LOS estimation on multiple links as a POMDP [43] since, the learning agent observes on the link(s) it selects for transmission. A POMDP is expressed as $\{\mathcal{S}, \mathcal{A}, P(s_{t+\Delta t}|s_t, a_t), r_t\}$, where $\mathcal{S}$ and $\mathcal{A}$ are state space and action space respectively while $P(s_{t+\Delta t}|s_t, a_t)$ is a transition probability from state $s_t \in \mathcal{S}$ to state $s_{t+\Delta t} \in \mathcal{S}$ and r_t is the immediate reward for the action a_t.

1) **State space**: We denote the state space as a set of vectors $\mathcal{S} = \{\mathbf{s}_t \mid \mathbf{s}_t = [s_{1t}, \dots, s_{Nt}]\}$, where N is the total number of available transmission links. In our case, $N = 2$. Since a channel can either be in LOS or NLOS state, we define a state of a channel as follows:

$$s_{nt} = \begin{cases} +1 & \text{if the } n^{th}\text{channel is in LOS} \\ -1 & \text{otherwise.} \end{cases}$$

2) **Agent's state Observation**: We denote the agent's observation of the channel states as a vector $\mathbf{\Omega}_t = [\omega_{1t}, \dots, \omega_{Nt}]$ where,

$$\omega_{nt} = \begin{cases} s_{nt} & \text{if the } n^{th}\text{channel is selected} \\ 0 & \text{otherwise} \end{cases}$$

3) **Action space**: An action carried out by the actor is the selection of a subset of the N channels. We, therefore, define the action space as a set of vectors $\mathcal{A} = \{\mathbf{a}_t \mid \mathbf{a}_t = [\lambda_{1t}, \dots, \lambda_{Nt}]\}$, where $\lambda_{nt} = 1$ indicates that the nth channel is selected and $\lambda_{nt} = 0$ otherwise, for $n = 1 \dots N$. Since in this study, N=2, $\mathcal{A} = \{[0,1], [1,0], [1,1]\}$.
4) **Reward**: The immediate reward r_t is expressed as a penalty whenever the E2E loss exceeds the defined threshold and is expressed as follows:

$$r_t = \begin{cases} \frac{-(\xi - \varrho)}{\rho} & \text{if } \xi \geq \varrho \\ \frac{1}{\rho} & \text{otherwise.} \end{cases} \tag{8}$$

TABLE 6: CTGAN and TVAE performance at different sizes of the training dataset

Size of the training dataset	**KL-Divergence**						**KS-test**						**Training Time (s)**	
	70°		60°		45°		70°		60°		45°			
	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE	CTGAN	TVAE
2k	0.0229	0.2751	0.0088	0.1356	0.7731	0.7731	0.9391	0.9479	0.8796	0.9711	0.1248	0.1248	0.85	0.55
5k	0.0003	0.2751	0.1356	0.1356	0.7731	0.7731	0.9952	0.9479	0.9711	0.9711	0.8752	0.8752	2.28	1.08
10k	**0.9297**	**0.2751**	**0.0256**	**0.1356**	**0.1837**	**0.4168**	**0.3617**	**0.9479**	**0.9763**	**0.9711**	**0.7255**	**0.8752**	**3.89**	**2.39**
20k	0.1874	0.2751	0.4077	0.1356	0.7731	0.7731	0.9485	0.9479	0.6183	0.9711	0.8752	0.8752	7.41	4.83
30k	0.2602	0.2751	0.0527	0.1356	0.0277	0.7731	0.9481	0.9479	0.9114	0.9711	0.9092	0.8752	14.17	8.21
40k	0.1816	0.2751	0.0221	0.1356	0.0227	0.7731	0.9485	0.9479	0.9773	0.9711	0.9183	0.8752	22.97	13.37
50k	0.0358	0.2751	0.0381	0.1356	0.0008	0.7731	0.9187	0.9479	0.9292	0.9711	0.9856	0.8752	24.32	15.46

where ξ represents the E2E loss evaluated over an episode, ϱ is the loss threshold, and ρ is the number of channels selected by the agent. When the loss is greater than the threshold, the first term motivates the agent to use multiple links to overcome the loss while the second term encourages the use of single link to conserve bandwidth in good channel conditions.

The basic architecture of the Actor-Critic RL consists of two networks: The actor and the critic as shown in Figure 6. The actor takes the action and the critic evaluates the action taken by the actor. In this work, we used two critic networks. The critic network calculates the current action-state value while the target-critic network computes the Bellman estimates of the future rewards. This approach improves the stability of the critic network because the target-critic is updated less often compared to the critic network. The three networks are updated according to equations (9), (10), and (12) respectively.

$$\nabla\phi_a J(\phi_a) = E_{\pi_{\phi_a}}[\nabla_{\phi_{at}} \ln\pi_{\phi_{at}}(\mathbf{s}_t, \mathbf{a}_t)\delta_t] \tag{9}$$

where $\nabla\phi_a J(\phi_a)$ is the policy gradient and $J(\phi_a)$ is the policy objective function.

$$Q^*_{\phi_c} = \arg\min_{Q_{\phi_c}}(\delta)^2 \tag{10}$$

where ϕ_a and ϕ_c are the actor and critic network parameters, and

$$\delta_t = r_t + \gamma Q_{\phi_c}(\mathbf{s}_{t+\Delta t}, \mathbf{a}_{t+\Delta t}) - Q_{\phi_c}(\mathbf{s}_t, \mathbf{a}_t) \tag{11}$$

$$\phi_{tc} = \alpha\, \phi_{tc} + (1-\alpha)\phi_c. \tag{12}$$

The choice of the Actor-Critic (AC) was motivated by the fact that the AC algorithm does not require prior knowledge of the model underlying the transmission channel. The AC algorithm searches the optimal policy on a parametrized family of functions using a gradient-based approach. We designed the AC networks using fully connected multi-layer perceptron NN with TensorFlow-2 [44] and Keras [45] libraries. More design and simulation parameters are given in Table 3.

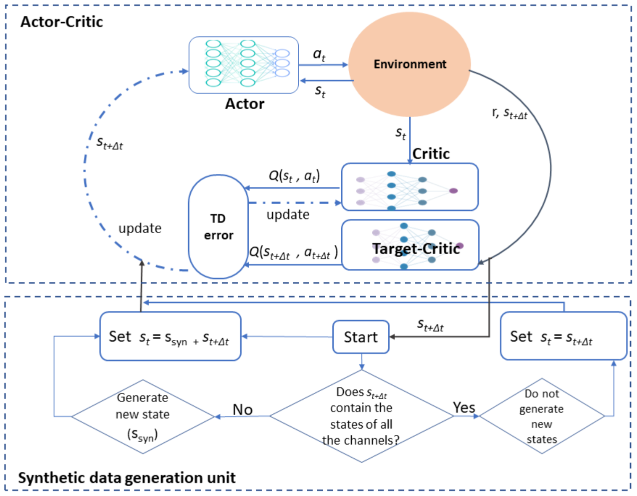

Actor-Critic
Actor
Environment
a_t
s_t
$r, s_{t+\Delta t}$
$s_{t+\Delta t}$
Critic
$Q(s_t, a_t)$
TD error
update
update
Target-Critic
$Q(s_{t+\Delta t}, a_{t+\Delta t})$
Set $s_t = s_{syn} + s_{t+\Delta t}$
Start
$s_{t+\Delta t}$
Set $s_t = s_{t+\Delta t}$
Generate new state (s_{syn})
No
Does $s_{t+\Delta t}$ contain the states of all the channels?
Yes
Do not generate new states
Synthetic data generation unit

FIGURE 6: The proposed learning architecture of the Actor-Critic Agent. The upper part is the architecture of the actor-critic algorithm while the lower part is the algorithm for generating synthetic data using generative models.

B. Transforming a POMDP into a FOMDP with GMs

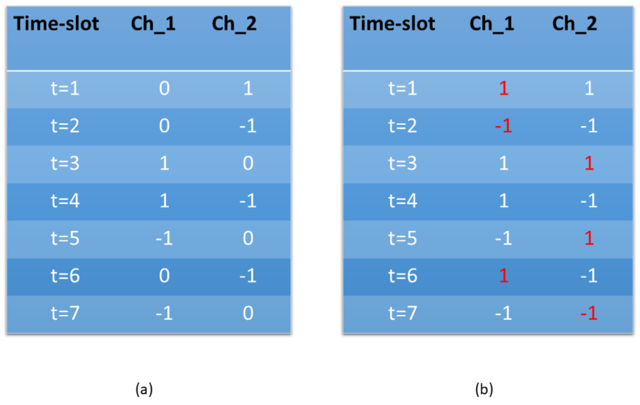
To accelerate the convergence of the RL agent, we propose the use of GMs to generate synthetic channel states for the states that are not accessed by the agent at a given time. This transforms the POMDP into a FOMDP and gives the agent complete knowledge of all channels. As a result, the agent converges faster to maximize the use of the satellite visibility period. As shown in Figure 7 (a), with POMDP, the agent only observes the states of the channels it selects for transmission, marked as 1 if the channel is in LOS and -1 if it is in NLOS, where 0 indicates that the channel was not accessed in that time slot and thus the agent has no state information for that channel. Whenever a channel is not accessed, its state is generated by the CTGAN and TVAE models. Therefore, the agent's observation of the channel states is modified as shown in Figure 7 (b), where the values in *red* mark the synthetic states generated by the GMs. In Figure 7 (b), the agent has a complete observation FOMDP of all states at any time slot. These state observations are fed as input to the actor-network, which learns the LOS probability of each channel and estimates the scheduling policy.

Algorithm 1: The Learning Process of the Actor-Critic Agent.

```
1:  Set $L$ as the total number of iterations, $M$ the episode length, and $N$ as the target-critic updating interval. Then, initialize the actor, critic, and target-critic networks with parameters $\phi_a$,$\phi_c$ $\phi_{tc}$ respectively.
2:  $\tau \leftarrow 0$;
3:  $l \leftarrow 0$;
4:  while $l \leq L$ do - The actor selects the action $\mathbf{a}_t \sim \pi_{\phi_a}(\mathbf{s}_t)$ i.e., the number of transmission links.
5:      $i \leftarrow 0$
6:      while $i \leq M$ do - Transmit the video on the selected links;
7:          if $i = M - 1$ then - Record the receiver report (channel states and loss rate) - Calculate the reward $r_t$ using (8); - The critic computes the state value; - Compute the TD error, $\delta_t$ using (11); - Update the actor and critic network parameters using (9) and (10) respectively:
8:              $\phi_a \leftarrow \phi_a(t)$
9:              $\phi_c \leftarrow \phi_c(t)$ - Update the Agent's Observation of the states according to Algorithm 2;
10:         end if
11:         $i \leftarrow i + 1$;
12:         $\tau \leftarrow \tau + 1$;
13:     end while
14:     if $\tau = N$ then - Update the target-critic network using (12);
15:         $\tau \leftarrow 0$;
16:     end if
17: end while
```

Algorithm 2: Generating Synthetic Channel States using CTGAN and TVAE

```
1:  Load the trained GMs into the simulation environment.
2:
3:  Initialize an array $\Omega$ of the Agent's state observation;
4:  $i \leftarrow 0$
5:  $\Omega \leftarrow s_{t+\Delta t}$ (future states)
6:  for $i \leq N$ do
7:      if $\Omega_t[i] = 0$ then
            - Generate synthetic state ($s_{syn}$) using CTGAN or TVAE model;
8:          $\Omega_t[i] \leftarrow s_{syn}$
9:      end if
10:     $i \leftarrow i + 1$
11: end for
```


(a)

Time-slot	Ch_1	Ch_2
t=1	0	1
t=2	0	-1
t=3	1	0
t=4	1	-1
t=5	-1	0
t=6	0	-1
t=7	-1	0

(b)

Time-slot	Ch_1	Ch_2
t=1	1	1
t=2	-1	-1
t=3	1	1
t=4	1	-1
t=5	-1	1
t=6	1	-1
t=7	-1	-1

FIGURE 7: The AC agent's Observation of the channel states. (a) POMDP and (b) FOMDP. The FOMDP is achieved by using synthetic channel states generated by the trained generative models. The link states are as follows: 1=LOS, -1= NLOS, red=synthetically generated channel state. In (a), 0 means the agent does not have the state of the respective link because the link was not accessed in a previous learning event.

C. Training Procedure

As detailed in Algorithm 1, at the start of an episode, the actor selects transmission links according to its observation Ω_t of the channel states and transmits the traffic on the selected links for the entire episode. At the end of the episode, the agent records the receiver report which contains the E2E loss rate for that episode and the state (LOS/NLOS) of each selected link determined by the last bit reception status. The E2E is used to calculate the reward which is then used by the critic and the target-critic to compute the current and future state values respectively. The Temporal Difference (TD) is found using (11) and is used to update both the critic and the actor networks. The agent's state observation is updated according to Algorithm 2. After a given number of iterations, the target critic is updated with a soft-update method; i.e., copying the weights of the critic network according to a defined update factor, which in our case is the learning rate of the critic. Figure 6 shows the schematic representation of the whole training procedure.

VII. Performance Evaluation

The goal of this study is to investigate whether converting a POMDP of the RL agent to a FOMDP by using GMs to generate synthetic channel states of the channels that the agent does not select can accelerate the convergence rate of the RL agent and allow it to transmit with the optimal policy for most of the satellite visibility time. To this end, we ran several simulations to train our agent in four different cases with different E2E loss thresholds: 0.0001, 0.0005, 0.001, and 0.01. In each of these four cases, the agent was trained using different channel states. First, we trained the agent in a partially observable Markov decision process (POMDP), where the agent learns from only the states of the channel it selects for transmission. Then we trained it in a fully observable Markov decision process (FOMDP) using real data obtained by using channel models. Finally, we trained the agent in FOMDP using synthetic data generated by CTGAN and TVAE models. In total, we ran 16 simulations to test the learning performance of our agent and the effect of using generative models. Each simulation lasted 1000 episodes with 1000 iterations for each episode. In the following, we evaluate the performance of the agent in terms of its learning performance, convergence rate, its ability to overcome the E2E loss, and the bandwidth used. For comparison, we also report the performance of the optimal scheduling policy. This is the policy that assumes that the LOS states of the channels are known in advance so that the steady-state probabilities for all available paths are exactly known.

A. Link Selection Performance

In this part, we evaluate the ability of our AC-agent to select suitable transmission links in various situations. Figure 8 compares the categorical distributions achieved at convergence by our AC-agent (red) and the optimal policy (blue). Categorical distributions are the probabilities of transmitting with satellite 1 (sat_1) at 70°, satellite 2 (sat_2) at 60°, and both satellites ($sat_{1,2}$).

1) Effect of E2E Loss Threshold

Four different E2E loss thresholds were considered: 0.0001, 0.0005,0.001,0.01. This means that in each case the learning agent has to determine the suitable links to use and whether to use single or double transmissions in order to overcome the E2E loss rate to meet the predefined threshold. It is expected that when the threshold is very low the agent should favor double transmissions compared to when the threshold is high. The results show that our agent is able to recognize this pattern and use double transmissions with redundancy when the E2E loss threshold is low at 0.0001 and 0.0005. Also when the threshold is moderate at 0.001, the agent still favors double transmission, but it also uses single transmission more than the previous two cases (0.0001 and 0.0005). However, when the E2E loss threshold is high at 0.01 the agent uses more single transmission because it is easy to meet the threshold without using redundancy to preserve bandwidth.

2) Effect of the Satellite Elevation Angle

Figure 8 shows that in the fourth case when the E2E loss threshold is 0.01 the agent uses single transmission and transmits more via satellite 1 than satellite 2 because satellite 1 is at a higher elevation angle of 70° compared to satellite 2 which is at 60°. Thus, satellite 1 is assumed to have a higher LOS probability than satellite 2 because, in urban areas, there are fewer obstacles like buildings at higher elevation angles than at lower elevation angles. These results show that our agent can learn the LOS probabilities of different links and select the suitable links that have higher LOS probability and higher chances of good traffic reception.

TABLE 7: Convergence Points of the AC-Agent in different Scenarios

Loss Threshold	Without GMs		With offline trained models		With models trained in real-time	
	POMDP	FOMDP with real data	FOMDP with CTGAN	FOMDP with TVAE	FOMDP with CTGAN	FOMDP with TVAE
0.0001	597	319	320	320	325	367
0.0005	532	210	224	209	225	220
0.001	246	20	25	43	96	84
0.01	4	3	3	3	4	4

3) Effect of using Generative Models

In each of the E2E loss thresholds considered, four different simulations were performed: using POMDP, FOMDP with real data, FOMDP with CTGAN model, and FOMDP with TVAE model as shown in Figure 8. The POMDP and FOMPD with real data are used as benchmarks to evaluate the effect of using synthetic data generated by CTGAN and TVAE models. It can be seen that in all four E2E loss thresholds, when the AC-agent uses FOMDP with synthetic data generated by CTGAN and TVAE, it achieves good performance similar to FOMDP with real data and outperforms the POMDP, especially in the fourth case when the E2E loss threshold is 0.01. This shows that using generative models to transform a POMDP into a FOMDP increases the learning performance of the AC agent in selecting suitable transmission links.

4) Comparison between the AC-Agent and the Optimal Policy

In Figure 8, the categorical distributions achieved by our agent are shown in red and those achieved by the optimal policy are shown in blue. The optimal policy is the scheduling policy that is assumed to have prior knowledge of the satellite LOS probabilities. The results show that in all the simulation scenarios considered, our learning agent achieves good performance comparable to the optimal policy which is assumed to know the channel states in advance.

B. Convergence Rate

In Table 7, we report the episodes in which the agent achieved convergence, i.e., the episode in which the KL - divergence between its categorical distributions and those of the optimal policy is minimal. These results show that for all E2E loss thresholds considered, using GMs to generate synthetic channel states increases the convergence rate of the learning agent compared to the case where the agent learns with partially observable channel states. For example, with an E2E loss threshold of 0.0001, the agent converges after 597 episodes in POMDP while it converges after

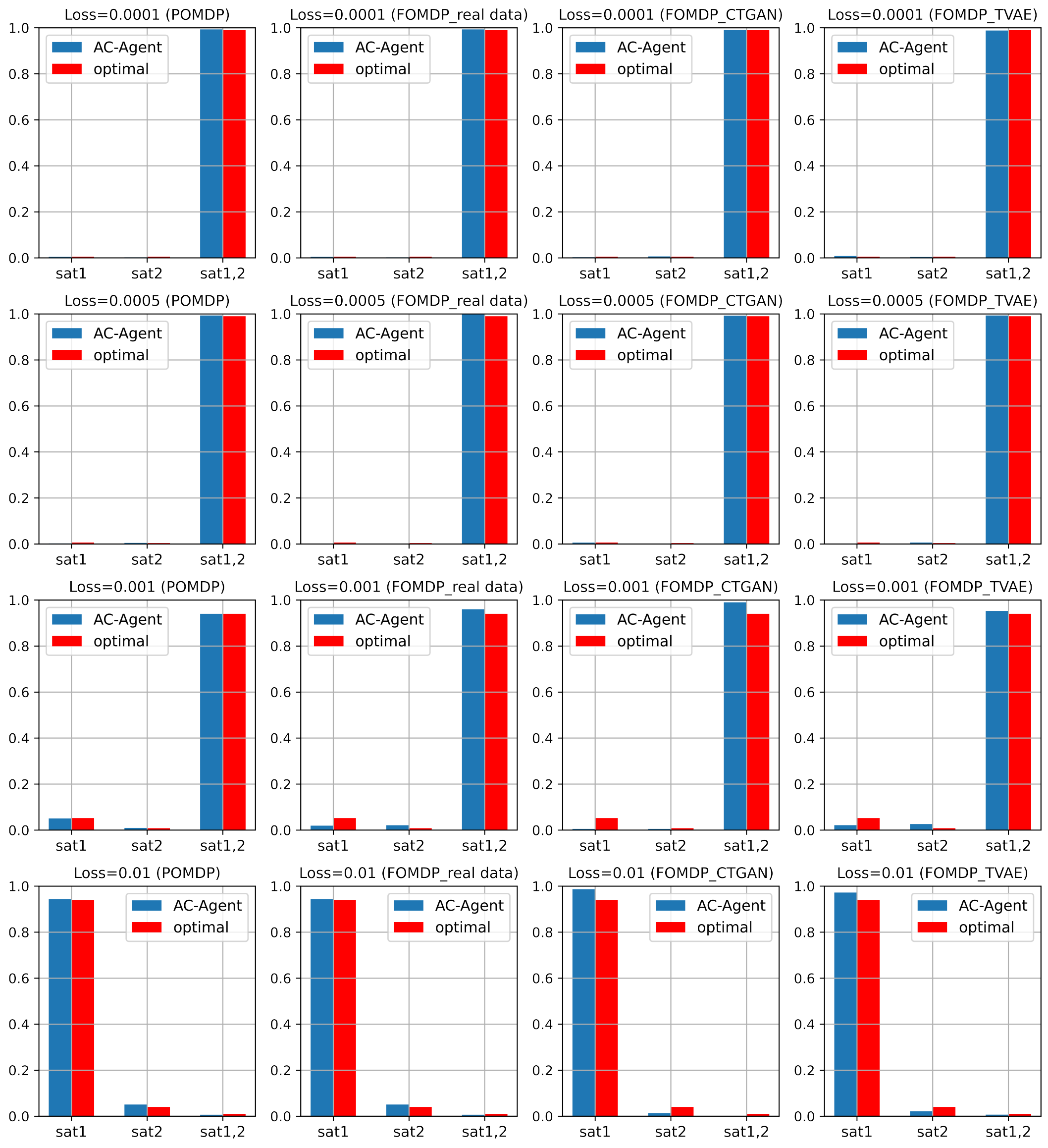


FIGURE 8: Comparison of the categorical distributions achieved by the AC-agent (red) and the optimal policy (blue), i.e., the probabilities to transmit with satellite 1 (sat1) at 70°, satellite 2 (sat2) at 60° or both satellites (sat1,2). Four E2E loss thresholds were considered: 0.0001,0.0005, 0.001, and 0.01, each in four scenarios: POMDP and FOMDP with real data, CTGAN and TVAE.

320 episodes in FOMDP with CTGAN and TVAE. This corresponds to a 47% increase in the convergence rate, a performance similar to the benchmark FOMDP when the agent uses real datasets and converges after 319 episodes. Similar improvements in convergence rate can be observed in all other scenarios. These results show that using GMs to generate synthetic channel states of the channels not selected by the agent, thereby converting a POMDP to a FOMDP, significantly improves the convergence rate of the learning agent. Figure 9 and Table 7 also show that TVAE converges faster and is more stable than CTGAN; perhaps because TVAE directly learns the distribution of the input data, unlike CTGAN. We also find that as E2E loss threshold increases, the agent converges relatively faster and arrives at a relatively better steady-state policy. This shows that our agent can operate in a wide range of propagation environments with different QoS requirements. The results in Table 7 also show that both offline and real-time-trained GMs achieve comparable performance. For example, at the 0.0001 E2E threshold, using real-time trained CTGAN and TVAE models the learning agent converges after 325 and 367 episodes, only 6 and 48 episodes respectively higher than with the offline-trained models. It can be concluded that our proposed approach of using GMs to accelerate the convergence of

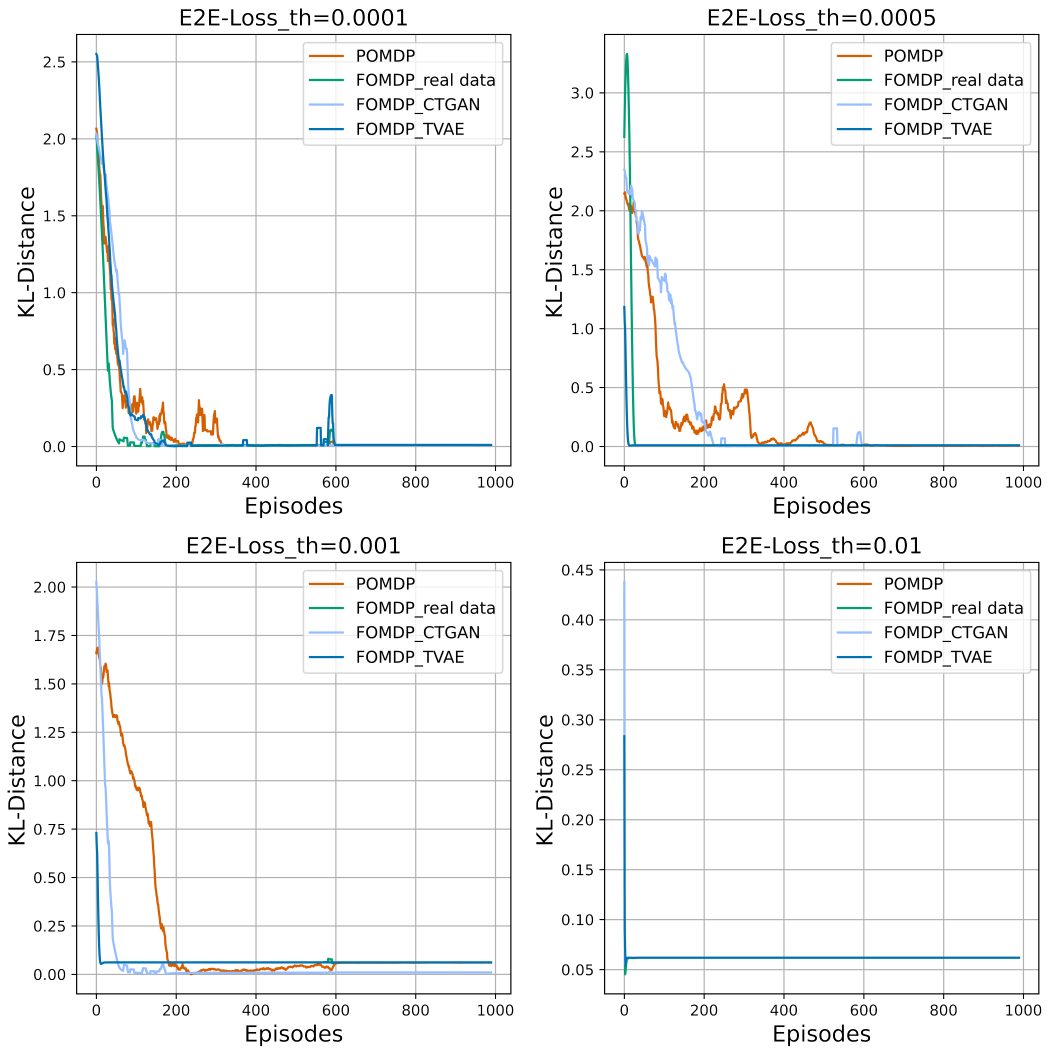


FIGURE 9: KL-Distance between the categorical distributions achieved by the AC-Agent and the optimal policy at different learning episodes to show the convergence rate of the AC-agent at different E2E loss thresholds:0.0001, 0.0005, 0.001, 0.01 in four scenarios: POMDP and FOMDP with real data, CTGAN and TVAE. KL-Distance decreases as the AC-agent converges to the optimal policy.

the learning agent can also be used in real-time operations without slowing down the agent's rate of convergence.

C. Convergence Time and the Satellite Visibility Period

In this part, we compare the agent convergence time and satellite visibility period. The results in Table 7 show that for partially observable states, the maximum convergence time is reached with a 0.0001 threshold and the agent converges after 597 episodes, which corresponds to 597k iterations. One iteration corresponds to the transmission of one bit. Assuming that the data rate of both the transmitter and receiver is 1 Mbps, which is feasible for satellite communications, convergence takes about 1.194 seconds, considering the time to transmit and receive feedback. This is about 0.001 times the satellite visibility period of 210 seconds in the considered scenario. In the case of the real-time trained- GMs, the training dataset of 10k was used, which can be obtained online in only 0.02 seconds, taking into account the transmission and feedback time. Table 6 shows that the CTGAN and TVAE models are trained in 3.89 and 2.39 seconds, respectively. Table 7 shows that when using real-time, the agent converges after 325 episodes with CTGAN and 367 episodes with TVAE, which correspond to 0.65 and 0.734 seconds, respectively. This means that the total time required to acquire the training data and train the GMs, as well as the time required for the learning agent to converge, is approximately 4.56 seconds for CTGAN and 3.144 seconds for TVAE, corresponding to 0.02 and 0.01 times the satellite visibility period respectively. It can be concluded that the learning agent can converge fast enough to make the best use of the satellite visibility time even when the GMs are trained in real-time.

Figure 9 is the graphical representation of the convergence rates of the learning agent in different scenarios. At the end of each learning episode, we recorded the KL-Distance between the categorical distributions achieved by the AC-agent and those achieved by the optimal policy. The results in Figure 9 show that in all the scenarios described above, the KL-Distance decreases as the simulation progresses. This shows that our agent is able to learn the LOS of the channels and converge to a steady-state scheduling policy when the KL -Distance approaches 0, i.e., the learning agent achieves the same categorical distributions as the optimal policy. It can also be seen that using generative models, in this case, CTGAN and TVAE, accelerates the convergence. This shows that our proposed approach of using GMs to transform a POMDP into a FOMDP can enable the learning agent to converge faster within the satellite visibility period.

D. E2E Loss Rate

In multipath transmission, traffic is considered lost if the transmitted traffic cannot be recovered on any of the available paths. In our case, the loss granularity is the bit. Therefore, the E2E loss rate (in BER) is defined as follows:

$$\mathcal{L} = \frac{\sum_{i=1}^{M} \upsilon}{\sum_{i=1}^{M} \omega} \tag{13}$$

where M is the number of iterations in an episode, and ω and υ are the number of bits transmitted and lost per iteration respectively.

Figure 10 shows the E2E loss rates achieved by the learning agent compared to the optimal policy at each of the loss thresholds:0.0001, 0.0005, 0.001, and 0.01 in different scenarios: POMDP and FOMDP with real data, CTGAN and TVAE. These results show that in all scenarios, as the agent continues to learn, the E2E losses decrease toward the end of the simulation. It can be seen that as intended, using GMs lowers the E2E loss more than using partially observable states, as the agent converges faster and transmits with the best policy most of the time. We also observe that TVAE shows better performance than CTGAN. Table 8 shows the numerical values of the average E2E loss rates. The high loss rates observed may be due to the fact that the reference city has satellite links with high losses due to low LOS probabilities, as shown in Table 1. For this reason, even with the optimal policy, the loss rate is higher than the thresholds, except for the highest threshold of 0.01.

E. Bandwidth Utilization

Figure 11 shows the bandwidth used in terms of the average number of bits transmitted by the learning agent and the optimal policy in each learning episode. It can be seen that at low loss thresholds (0.0001, 0.0005, and 0.001), both the learning agent and the optimal policy trade the bandwidth to overcome the E2E loss. While the optimal policy uses double transmission most of the time, the learning agent starts with single transmission and slowly learns and converges towards double transmissions. However, at the higher loss threshold of 0.01, both converge to a single transmission. Results in Figure 8 too, show this behavior of using high bandwidth at low loss thresholds and low bandwidth at high loss thresholds. Table 9 shows the average throughput in megabits per second (Mbps). In our simulations, we assumed 1.5 Mbps as the source rate. These results show that our agent can learn the link characteristics and proactively transmit with redundancy to overcome high losses and use single transmission in low-loss conditions to save bandwidth.

TABLE 8: E2E Loss Rate (BER) in different scenarios

Loss Threshold	POMDP	FOMDP with real data	FOMDP with CTGAN	FOMDP with TVAE	Optimal Policy
0.0001	0.0265	0.0258	0.0260	0.0219	0.0049
0.0005	0.0265	0.026	0.0252	0.0260	0.0048
0.001	0.0201	0.0242	0.028	0.0222	0.0056
0.01	0.022	0.021	0.036	0.035	0.017

TABLE 9: Bandwidth used in different scenarios

Loss Threshold	POMDP	FOMDP with real data	FOMDP with CTGAN	FOMDP with TVAE	Optimal Policy
0.0001	2.3265	2.3265	2.325	2.3715	2.985
0.0005	2.319	2.34	2.31	2.355	2.97
0.001	2.289	2.34	2.289	2.355	2.91
0.01	1.833	1.8	1.815	1.8	1.515

VIII. Conclusion

In this work, we presented an AI-based framework for the upcoming 6G-NTN integrated networks. The framework consists of an AC-RL agent and GMs. The RL agent estimates the LOS probabilities and schedules traffic over multiple access links connecting the UE to LEO satellites in a multi-access mode, while the generative models (GANs and VAEs) are used to transform a POMDP into a FOMDP to accelerate the learning process of the agent so that it can converge within the satellite visibility period. Simulation results have shown that our approach significantly improves the learning process and shortens the convergence time. As a result, the agent is able to transmit with an optimal policy for most of the satellite visibility period, thus satisfying the QoS requirements by reducing E2E losses without incurring additional bandwidth costs. In the 6G context, our framework can offer learning-based LOS estimation and traffic scheduling in 6G-NTN integrated networks to improve link reliability and availability, increase data rates and throughput, and improve the QoS and the user Quality of Experience (QoE) which are among the main pillars of the upcoming 6G mobile networks. In addition, we have shown that the GMs can be trained in real-time using network data collected by the RL agent, eliminating the need for prior knowledge of the channel model or training data.

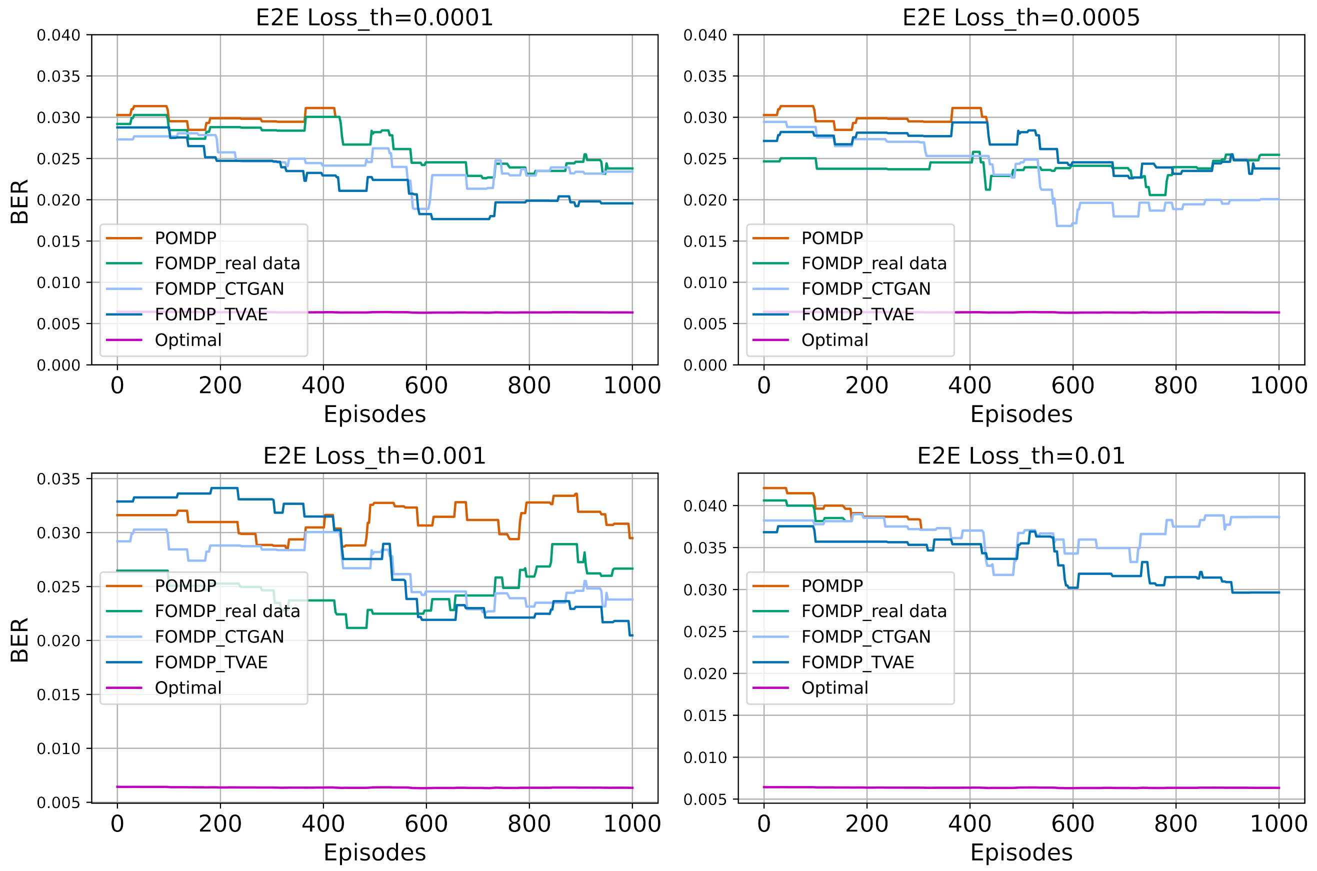


FIGURE 10: E2E loss rate achieved by the AC-Agent and the Optimal policy at different E2E loss thresholds: 0.0001, 0.0005, 0.001, 0.01 in four scenarios: POMDP and FOMDP with real data, CTGAN and TVAE.

Acknowledgments
This work was supported by the European Union under the Italian National Recovery and Resilience Plan (PNRR) PE00000001 - program ”RESTART”, the European Union under the Italian National Recovery and Resilience Plan (PNRR) Mission 4 Issue 2 Investment 1.4 “Potenziamento strutture di ricerca e creazione di ”campioni nazionali di R&S” CN00000023 – “Sustainable Mobility Center (CNMS)” and by the HORIZON-CL4-2021-SPACE-01 project ”5G+ evoluTion to mutioRbitAl multibaNd neTwORks” (TRANTOR) No. 101081983

REFERENCES
[1] ITU-R-WP5D, “Future technology trends of terrestrial international mobile telecommunications systems towards 2030 and beyond,” *International Telecommunication Union*, 2022. [Online]. Available: https://www.itu.int/pub/R-REP-M.2516

[2] R. Liu, R. Y.-N. Li, M. Di Renzo, and L. Hanzo, “A vision and an evolutionary framework for 6g: Scenarios, capabilities and enablers,” *arXiv preprint arXiv:2305.13887*, 2023.

[3] J. Du, C. Jiang, J. Wang, Y. Ren, and M. Debbah, “Machine learning for 6g wireless networks: Carrying forward enhanced bandwidth, massive access, and ultrareliable/low-latency service,” *IEEE Vehicular Technology Magazine*, vol. 15, no. 4, pp. 122–134, 2020.

[4] 3GPP, “Technical specification group radio access network; solutions for nr to support non-terrestrial networks (ntn): Tr 38.821 v16.1.0 (2021-05), (release 16),” 2021.

[5] M. Latva-aho, K. Leppänen, F. Clazzer, and A. Munari, “Key drivers and research challenges for 6g ubiquitous wireless intelligence,” *6G Flagship white paper*, 2020.

[6] X. Zhu and C. Jiang, “Integrated satellite-terrestrial networks toward 6g: Architectures, applications, and challenges,” *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 437–461, 2021.

[7] D. Zhou, S. Gao, R. Liu, F. Gao, and M. Guizani, “Overview of development and regulatory aspects of high altitude platform system,” *Intelligent and converged networks*, vol. 1, no. 1, pp. 58–78, 2020.

[8] E. Juan, M. Lauridsen, J. Wigard, and P. E. Mogensen, “A time-correlated channel state model for 5g new radio mobility studies in leo satellite networks,” in *2021 IEEE 93rd Vehicular Technology Conference (VTC2021-Spring)*, 2021, pp. 1–5.

[9] 3GPP, “Technical specification group services and system aspects; system architecture for the 5g system (5gs): Ts 23.501 v18.1.0 (2023-03), (release 18),” 2023.

[10] A. Machumilane, A. Gotta, P. Cassará, G. Amato, and C. Gennaro, “Learning-based traffic scheduling in non-stationary multipath 5g non-terrestrial networks,” *Remote Sensing*, vol. 15, no. 7, p. 1842, 2023.

[11] J. D. Williams and S. Young, “Partially observable markov decision processes for spoken dialog systems,” *Computer Speech & Language*, vol. 21, no. 2, pp. 393–422, 2007.

[12] L. Xu, M. Skoularidou, A. Cuesta-Infante, and K. Veeramachaneni, “Modeling tabular data using conditional gan,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[13] A. Badr, A. Khisti, W.-T. Tan, and J. Apostolopoulos, “Perfecting protection for interactive multimedia: A survey of forward error correction for low-delay interactive applications,” *IEEE Signal Processing Magazine*, vol. 34, no. 2, pp. 95–113, 2017.

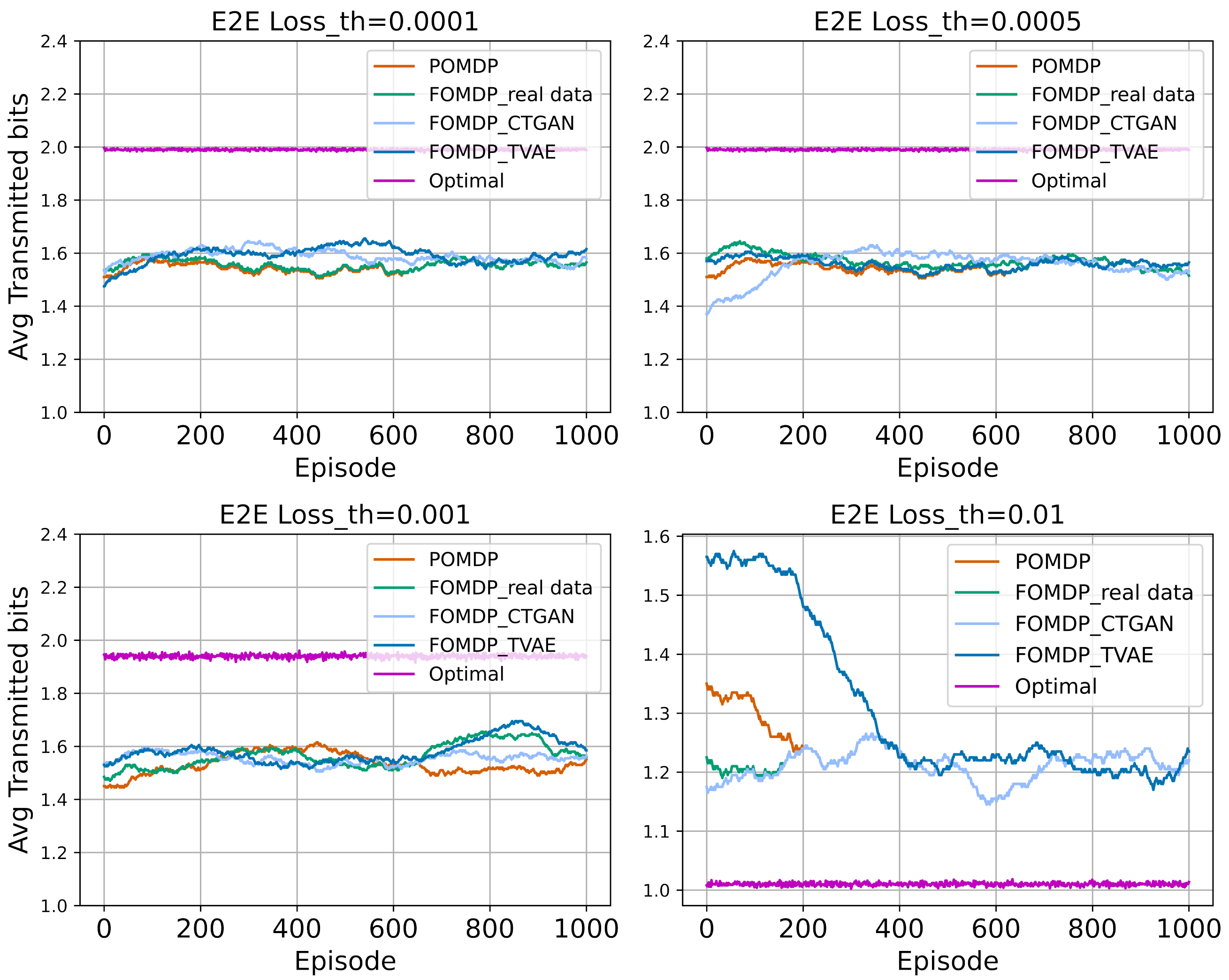


FIGURE 11: Bandwidth used by the AC-Agent and the Optimal policy in terms of the average transmitted bits in each learning episode at different E2E loss thresholds: 0.0001, 0.0005, 0.001, 0.01 in four scenarios: POMDP and FOMDP with real data, CTGAN and TVAE.

[14] Y. Sun and L. Fu, "Stacking ensemble learning for non-line-of-sight detection of global navigation satellite system," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022.
[15] V. Aydın, İ. H. Çavdar, and Z. HASIRCI, "Line of sight (los) probability prediction for satellite and haps communication in trabzon, turkey," *International Journal of Applied Mathematics Electronics and Computers*, no. Special Issue-1, pp. 155–160, 2016.
[16] C. Paasch, S. Ferlin, O. Alay, and O. Bonaventure, "Experimental Evaluation of Multipath TCP Schedulers," in *ACM SIGCOMM workshop on Capacity sharing workshop*, 2014, pp. 27–32.
[17] J. Wu, C. Yuen, B. Cheng, Y. Shang, and J. Chen, "Goodput-Aware Load Distribution for Real-Time Traffic over Multipath Networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 8, pp. 2286–2299, 2014.
[18] P. Houze, E. Mory, G. Texier, and G. Simon, "Applicative-Layer Multipath for Low-Latency Adaptive Live Streaming," in *International Conference on Communications (ICC)*. IEEE, 2016, pp. 1–7.
[19] S. Afzal, C. E. Rothenberg, V. Testoni, P. Kolan, and I. Bouazizi, "Multipath MMT-based Approach for Streaming High Quality Video over Multiple Wireless Access Networks," *Computer Networks*, vol. 185, pp. 1–18, 2021.
[20] Q. Wang, T. Nguyen, and B. Bose, "Towards adaptive packet scheduler with deep-q reinforcement learning," in *2020 International Conference on Computing, Networking and Communications (ICNC)*, 2020, pp. 118–123.
[21] H. Wu, Ö. Alay, A. Brunstrom, S. Ferlin, and G. Caso, "Peekaboo: Learning-based multipath scheduling for dynamic heterogeneous environments," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 10, pp. 2295–2310, 2020.
[22] C. Zhong, Z. Lu, M. C. Gursoy, and S. Velipasalar, "A deep actor-critic reinforcement learning framework for dynamic multichannel access," *IEEE Transactions on Cognitive Communications and Networking*, vol. 5, no. 4, pp. 1125–1139, 2019.
[23] H. Yang and X. Xie, "An actor-critic deep reinforcement learning approach for transmission scheduling in cognitive internet of things systems," *IEEE Systems Journal*, vol. 14, no. 1, pp. 51–60, 2019.
[24] Y. Wei, F. R. Yu, M. Song, and Z. Han, "Joint optimization of caching, computing, and radio resources for fog-enabled iot using natural actor–critic deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 2061–2073, 2018.
[25] A. Machumilane, A. Gotta, P. Cassará, C. Gennaro, and G. Amato, "Actor-critic scheduling for path-aware air-to-ground multipath

multimedia delivery," in *2022 IEEE 95th Vehicular Technology Conference:(VTC2022-Spring)*. IEEE, 2022, pp. 1–5.

[26] A. Gotta and P. Barsocchi, "Experimental video broadcasting in dvb-rcs/s2 with land mobile satellite channel: a reliability issue," in *2008 IEEE International Workshop on Satellite and Space Communications*. IEEE, 2008, pp. 234–238.

[27] N. Celandroni and A. Gotta, "Performance analysis of systematic upper layer fec codes and interleaving in land mobile satellite channels," *IEEE Transactions on Vehicular Technology*, vol. 60, no. 4, pp. 1887–1894, 2011.

[28] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[29] K. Sarkar, L. Liu, V. Golyanik, and C. Theobalt, "Humangan: A generative model of human images," in *2021 International Conference on 3D Vision (3DV)*, 2021, pp. 258–267.

[30] I. Durugkar, I. Gemp, and S. Mahadevan, "Generative multi-adversarial networks," *arXiv preprint arXiv:1611.01673*, 2016.

[31] Y. Li, M. Min, D. Shen, D. Carlson, and L. Carin, "Video generation from text," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, Apr. 2018. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/12233

[32] D. Rajan and J. J. Thiagarajan, "A generative modeling approach to limited channel ecg classification," in *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. IEEE, 2018, pp. 2571–2574.

[33] W. Xia, S. Rangan, M. Mezzavilla, A. Lozano, G. Geraci, V. Semkin, and G. Loianno, "Generative neural network channel modeling for millimeter-wave uav communication," *IEEE Transactions on Wireless Communications*, vol. 21, no. 11, pp. 9417–9431, 2022.

[34] S. Bano, A. Machumilane, P. Cassarà, and A. Gotta, "How generative models improve los estimation in 6g non-terrestrial networks," 2023.

[35] S. Bano, P. Cassarà, N. Tonellotto, and A. Gotta, "A federated channel modeling system using generative neural networks," 2023.

[36] H. Ye, L. Liang, G. Y. Li, and B.-H. Juang, "Deep learning-based end-to-end wireless communication systems with conditional gans as unknown channels," *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3133–3143, 2020.

[37] J. C. McDowell, "The low earth orbit satellite population and impacts of the spacex starlink constellation," *The Astrophysical Journal Letters*, vol. 892, no. 2, p. L36, apr 2020. [Online]. Available: https://dx.doi.org/10.3847/2041-8213/ab8016

[38] Recommendation-ITU-R, "Propagation data required for the design of earth-space land mobile telecommunication systems," *International Telecommunication Union*, pp. 681–10, 2017.

[39] E. Lutz, D. Cygan, M. Dippold, F. Dolainsky, and W. Papke, "The land mobile satellite communication channel-recording, statistics, and channel model," *IEEE transactions on vehicular technology*, vol. 40, no. 2, pp. 375–386, 1991.

[40] H. Bischel, M. Werner, and E. Lutz, "Elevation-dependent channel model and satellite diversity for ngso s-pcns," in *Proceedings of Vehicular Technology Conference-VTC*, vol. 2. IEEE, 1996, pp. 1038–1042.

[41] J. F. Nash Jr, "Equilibrium points in n-person games," *Proceedings of the national academy of sciences*, vol. 36, no. 1, pp. 48–49, 1950.

[42] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *CoRR*, vol. abs/1312.6114, 2013.

[43] G. E. Monahan, "State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms," *Management science*, vol. 28, no. 1, pp. 1–16, 1982.

[44] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, R. Jozefowicz, Y. Jia, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, M. Schuster, R. Monga, S. Moore, D. Murray, C. Olah, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow, Large-scale machine learning on heterogeneous systems," November 2015.

[45] F. Chollet *et al.*, "keras," https://github.com/fchollet/keras, 2015.