[1]Ishwari V. Ginimav

[2]Gowrishankar S.

[3]Ramesh Babu H. S

[4]Prasad G. R.

DLLBVS: Design of a High-Efficiency Deep Learning based Low-BER Video Streaming Model for High-Noise Wireless Networks

Abstract: - Design of video streaming models for high-noise networks is a multimodal process that requires efficient channel modelling, noise analysis, error-specific stream control, intelligent data augmentation, etc. This article proposes design of a high-efficiency deep learning based low-BER (Bit Error Rate) Video streaming model for high-noise wireless networks. The proposed model initially uses a Long-Short-Term Memory (LSTM) block for estimation of frame level features, and then uses Orthogonal Frequency Division Multiple Access (OFDMA) based modulation platform for transmission and reception of processed video frames. The OFDMA model is cascaded with a chaotic communication module, which assists in improving data fidelity under different noise conditions. The chaotic communication module is optimized using Grey Wolf Optimizer (GWO), which assists in setting up its hyperparameters for better efficiency under real-time communication scenarios. Video frames were pre-processed using Dual Neural Networks that assisted in estimation of differential frame information sets. Due to estimation of this differential frame information, streaming speed is improved, which assists in increasing number of frames transmitted per second, thereby improving streaming performance for different video types. This frame information is further processed by an iterative Gated Recurrent Unit (GRU) based VARMAx Model, which assists in predicting frame removal instances, thus assisting in faster & low error communications. The GRU VARMAx Model optimizes data flow, thus reducing congestion and improving throughput. The model was tested under AWGN (Additive While Gaussian Noise), Rayleigh, & Rician channel types, and its efficiency was compared with standard streaming techniques in terms of communication delay, Bit Error Rate, Peak to Average Power Ratio (PAPR), throughput, communication jitter and computational complexity levels. Based on this comparison it was observed that the proposed model showcased 3.5% lower communication delay, 8.3% lower BER, 5.9% lower PAPR, 10.5% higher throughput, 3.9% lower jitter and 6.4% lower computational complexity, which makes it highly useful for a wide variety of real-time streaming scenarios.

Keywords: OFDMA, Dual Neural Network, LSTM, BER, PAPR, Delay, GRU, VARMAx, GWO, Throughput

I. INTRODUCTION

The IEEE 802.16 standard may be able to enable video streaming models with different traffic flows and changing channel characteristics. Due to the implementation of complex antenna techniques and adjustable sub channelization, the system is able to deliver peak downlink (DL). This permits wide use of the system to offer both fixed and mobile services for business and consumer markets. Recent advancements in domains such as high-speed Internet, multimedia, and video broadcasting applications have rendered WiMAX indispensable. Work in [1] describes an end-to-end, non-hierarchical WiMAX network sets. This has the potential to be expanded to execute optional relay entities, hence enhancing coverage and performance in future versions. The IEEE 802.16 standard specifies the Medium-Access-Control (MAC) and Physical Layer (PHY) protocols for fixed and mobile Broadband wireless-access systems. The MAC functions and physical functions may be categorized into the data plane, the control plane, and the management plane, respectively. The data plane consists of the functions that comprise the data processing pipeline. Examples include compression of the header, MAC and PHY components, and functions for processing data packets.

A collection of layer-2 (L2) control functions is necessary to enable a range of radio resource configurations, as well as Coordination, Signalling, and Management. The community colloquially refers to this collection of functions

[1] Department of Computer Science and Engineering, B.M.S. College of Engineering, Bengaluru, 560019, India

[2]Deparment of Computer Science and Engineering, B.M.S. College of Engineering, Bengaluru, 560019, India

[3]Principal, Sai Vidya Institute of Technology, Doddaballapur Road, Rajanukunte, Bengaluru, 560064, India

[4]Deparment of Computer Science and Engineering, B.M.S. College of Engineering, Bengaluru, 560019, India

[1]ishwari@bmsce.ac.in

[2]gowrishankar.cse@bmsce.ac.in

[3]rbabuhss@gmail.com

[4]prasad.cse@bmsce.ac.in

as the control-plane functions. Additionally, a management plane that can be utilized for both internal and external system settings is created. WiMAX Architecture is adaptable for usage in a range of applications, and the specification supports two basic exploitation scenarios: Last-mile Broadband Wireless Access: In the present context, broadband wireless connection acts as the interface between residential and business WMAN customers. The functionality requires a point-to-multipoint single-hop transmission, a single base station (BS), and several subscriber stations to function successfully (SSs). Backhaul systems: Using meshing among IEEE 802.16/WiMAX Subscriber Stations, a WiMAX network may transmit data and voice traffic from the cellular edge to the core network (the Internet). It is believed that the collection, processing, and storage of data are the most essential parts of telemedicine systems [2, 3, 4]. Personal medical information system servers, commonly known as PMIS servers, are used to store diagnostic and personal patient information inside the system. These servers are connected to the hospital's network, allowing the medical staff local and remote access to the data they contain.

In recent years, telemedicine services with applications in emergency healthcare, telecardiology, teleradiology, telepathology, tele-dermatology, and tele-oncology

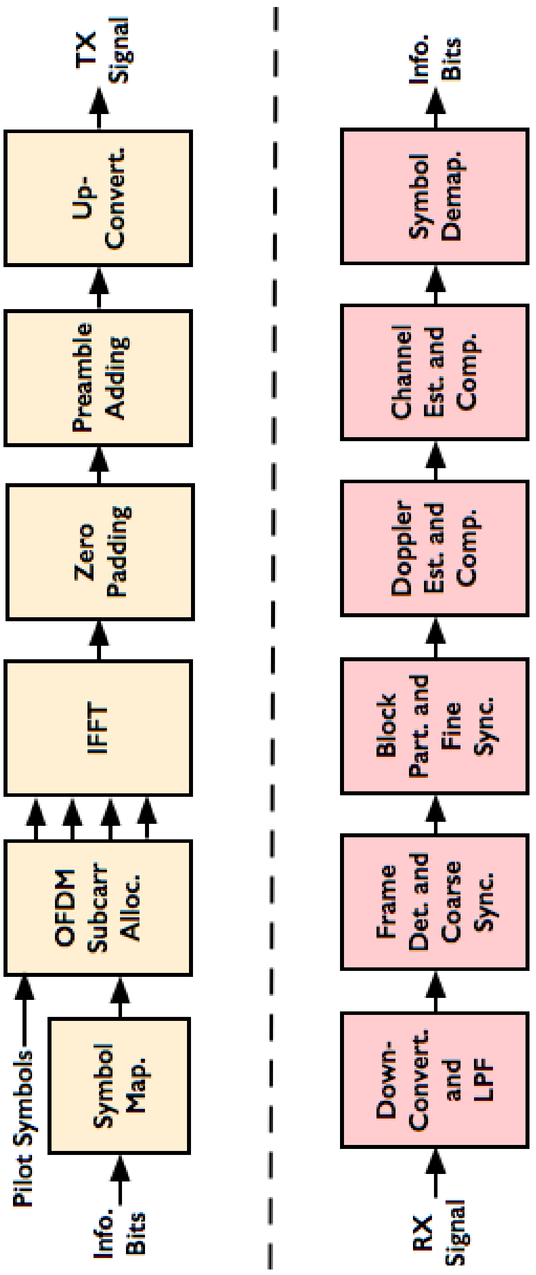

Figure 1. Design of the OFDM based communication interface sets

have gained popularity as a way of delivering fast and effective patient care. IEEE 802.16/WiMAX technology offers several advantages for wireless telemedicine and e-health services, including high bandwidth, security, excellent quality of service, and integrated service provision. The IEEE 802.16/WiMAX standard's media access control (MAC) layer security feature, as represented in figure 1, has the ability to offer encryption and access control features for wireless telemedicine services. The only two layers described by the IEEE 802.16e standard are the physical (PHY) layer and the medium access control (MAC) layer. The Physical Layer (PHY) is responsible for the actual data transmission onto the Air Interface (AI) between Peer Units, while the Media Access Control (MAC) Layer is responsible for establishing and maintaining the connection between communication devices. PHY layer functions for real-time applications include signal encoding and decoding, production of the preamble, modulation and demodulation, transmission and receipt of bits, and management of transmission power under different use cases.

Such models are capable of communicating in both line-of-sight (LOS) and non-line-of-sight (NLOS) modes utilizing a pre-set range of frequencies. OFDM, which stands for "Orthogonal Frequency Multiplexing," is one of the modulation techniques created for WiMAX to prevent ISI and power loss. The basis of orthogonal frequency division multiplexing (OFDM) is frequency division multiplexing (FDM), which uses several frequencies to transmit signals in parallel. The choosing of carrier spacing is inversely related to symbol time. Thanks to its design, the MAC layer of IEEE 802.16 supports PMP and mesh networks. In mesh networks, subscriber stations (SS) communicate directly with one another without going via base stations (BS) (Base station). It provides connection-oriented service construction to the higher-tier service. Each service connection is issued a unique 16-bit Connection ID before it is permitted to reach the MAC layer (CID). At the CS layer, two unique events occur: first, the CS layer classifies the incoming packet based on the IP protocol and priority criteria, and then it transfers the packet to the appropriate CID. The unidirectional connections between the BS and SS may both control the air interface and transmit service flow traffic sets.

Similar models that execute streaming operations are reviewed in the next section, where it was noticed that the majority of them are either very sophisticated or have static settings, limiting their real-time deployment potential. In addition, several of these models include deployment-specific settings, which diminishes their scalability when applied to large-scale networks with hybrid noise types. Section 3 of this paper proposes the creation of a low-BER (Bit Error Rate) Video streaming model for high-noise wireless networks based on high-efficiency deep learning. The performance of the suggested model was assessed using several quantitative indicators and compared to typical streaming approaches. Finally, this paper concludes with some context-specific observations regarding the proposed model, as well as suggestions for optimizing its performance in various use case scenarios.

Limitations of existing models

In the paper, we highlight two primary limitations prevalent in existing video streaming models: complexity and static settings. These limitations are explored in detail to elucidate their impact on the performance and applicability of current streaming technologies in high-noise wireless environments.

Firstly, the aspect of complexity in communication systems is critically analyzed. This complexity manifests through various dimensions, including technical, computational, and operational complexities. The technical complexity encompasses the use of advanced algorithms and sophisticated error-correction mechanisms, which, while enhancing performance, also increase the difficulty in implementation and maintenance. Computational complexity, another facet of this issue, refers to the high processing power required by these systems, posing a challenge in resource-constrained environments. Furthermore, operational complexity indicates that these intricate systems demand specialized knowledge for effective management, thus complicating troubleshooting and adaptation processes.

Secondly, the authors discuss the limitations imposed by static settings in communication systems. Static settings imply a rigidity in the system configuration, lacking the necessary flexibility to adapt to the dynamic nature of wireless networks. Such systems, optimized for a specific set of conditions, often falter in the face of varying noise levels, traffic volumes, and other changing parameters. This inflexibility hinders scalability, rendering these systems less effective in expanded network scenarios. Additionally, static systems are at risk of becoming obsolete as they may not evolve in tandem with technological advancements and changing user demands.

The paper then contrasts these identified limitations with the proposed model, emphasizing how the new design addresses and mitigates these issues for real-time scenarios. The model's adaptability to diverse noise conditions and its capability to efficiently handle various video types and network scenarios are highlighted. This contrast distinctly positions the proposed model as a significant advancement in the domain of video streaming within high-noise wireless networks, offering a solution that is both innovative and practical in addressing the challenges faced by contemporary streaming technologies.

II. LITERATURE REVIEW OF EXISTING STREAMING MODELS

A wide variety of deep-learning based models are proposed for processing of video sets. For instance, work in [5, 6] propose use of Two-Stream Convolutional Network (TSCN), and virtual transcoder function (VTF) for processing video sets. However these models do not incorporate streaming operations, which limits their scalability levels. To overcome this issue, work in [7] proposes use of Perceptually Optimized Adaptive Video Streaming, which assists in pre-emption of video sequences for efficient transmission over different network scenarios. Similar models are proposed in [8, 9, 10], which discuss use of tile-view graphs (TVGs), Dynamic Quality Boundaries (DQB), and Log-Rectilinear Transformation (LRT), via which high-density feature sets are extracted, and communicated over dense networks. Extensions to these models are discussed in [11, 12, 13], where researchers have proposed use of vectorised policy gradient model, DASH based edge computing model, and Linear Quadratic Regulator (iLQR) are deployed under real-time use cases. These models are capable of extracting high variance frame sets, that can be deployed for improving Quality of Service (QoS) under different streaming scenarios.

Models that use Deep Reinforcement Learning (DRL) [14], Linear Optimizations [15], Online Bitrate Selection [16], Deep Learning Optimizations [17], cycle vector-quantized variational autoencoder (cycle-VQ-VAE) [18], Flexible Latency Aware Streaming (FLAS) [19], and Reinforcement Learning-Based Rate Adaptation (RLRA) [20], for dynamic control over streaming operations are discussed & evaluated under different scenarios. These models are further extended via the work in [21, 22, 23, 24, 25], which propose use of Shift-Tile-Tracking (STC), LSTM based streaming, scalable-high-efficiency-video-coding (SHVC) with device-to-device communications, Sliding-Window Forward Error Correction (SW FEC), and context-aware streaming, which enables real-time processing for different video types. Extended models that use Proactive Caching [26], and Video streaming based on super resolution [27] are also discussed, and are highly useful for a wide variety of real-time use cases. But most of them are either highly complex, or have static configurations, which limits their real-time deployment capabilities.

Esakki et al. [1] make a substantial contribution to adaptive video encoding, focusing on different video codecs. Their work is pivotal in understanding how adaptive encoding can be optimized for various codecs, enhancing streaming quality and efficiency. Similarly, Barman and Martini [4] explore user-generated HDR gaming video streaming, providing insights into codec comparisons and the challenges associated with high-dynamic-range content. These studies underline the importance of codec selection and adaptation in modern video streaming.

Several studies have focused on enhancing the user quality of experience (QoE) in video streaming. Park et al. [3], Bampis et al. [7], and Hu et al. [8] delve into machine learning approaches and perceptual optimization to improve QoE in various streaming scenarios, including 360-degree video. These contributions are crucial in an era where viewer satisfaction is paramount, and streaming services seek to deliver increasingly immersive content.

The integration of machine learning and AI in video streaming has been a game-changer, as evidenced by the work of Hu et al. [5] in detecting compressed deepfake videos and Tang and Wong [16] in online bitrate selection for viewport adaptive streaming. These advancements show the potential of AI in enhancing streaming quality and security.

Efficient resource utilization is another critical area in video streaming, as highlighted by Erfanian et al. [6] in their study on optimizing resource utilization in live video streaming. The efficiency of streaming services not only impacts the viewer experience but also the overall network and server load, making this an important area of study.

Emerging trends in video streaming are also well represented in recent literature. For instance, the work of Li et al. [10] on a log-rectilinear transformation for foveated 360-degree video streaming and the research by Jiménez et al. [17] on the impact of uplink performance on 360° live video streaming in LTE networks shed light on new technologies and techniques that are shaping the future of video streaming. The challenges and future scopes in video streaming technology are also well documented. Studies like that of Cui et al. [14] on transmission control in live video streaming and Zheng et al. [21] on FoV tracking enabled high-quality VR video streaming on mobile platforms indicate ongoing efforts to address the evolving demands of video streaming, particularly in the context of virtual reality and mobile platforms.

Cheng et al. [26, 27, 28] and Zhang et al. [29, 30, 31, 43] have made significant contributions to the field of mobile VR video streaming. Cheng et al. focus on MEC- and proactive caching-based solutions, while Zhang et al. explore field-of-view (FoV) tracking for high-quality 16K VR video streaming on mobile platforms. These studies address the increasing demand for immersive and high-quality mobile VR experiences. The role of edge computing in enhancing video streaming experiences is evident in the works of Zhang et al. [27, 32, 33, 34] and Farahani et al. [35, 36 37]. Zhang et al. present an edge-assisted adaptive video streaming solution that incorporates video super-resolution and caching, while Farahani et al. introduce a collaborative edge-assisted framework for HTTP adaptive video streaming. These approaches highlight the potential of edge computing in reducing latency and improving the quality of streamed content.

The Quality of Experience (QoE) remains a central focus in video streaming, as demonstrated by Tanjung et al. [30, 38, 39, 40] and Li et al. [34, 41, 42, 43] in their respective studies on QoE optimization in DASH-based multiview video streaming and improving QoE for low-latency live video streaming. Additionally, Chakareski et al. [32, 44, 45] provide insights into end-to-end optimization for viewport-driven 360° video streaming, emphasizing the importance of user navigation modeling and rate-distortion analysis. Innovative approaches to video streaming are explored by Lee et al. [46, 47, 48] in their development of a group-based adaptive rendering system for 6DoF immersive video streaming, and by Yuan et al. [38,49, 50] in their work on multi-source streaming for tile-based 360° videos within cloud-native 5G networks. These studies showcase the integration of cutting-edge technologies to enhance the streaming experience.

Addressing challenges in video streaming, Gao et al. [44] introduce an intelligent video processing architecture, and Park et al. [40] propose a neural adaptive power consumption optimization for mobile video streaming. These contributions reflect the ongoing efforts to tackle the diverse challenges in video streaming, ranging from processing complexities to power consumption concerns. Emerging research in the field, such as the study by Avanzato et al. [45, 51, 52] on enhancing video quality in drone communications using VPN bonding, and the work of Lee et al. [47] on space-time subsampled video quality, offer new perspectives and future directions for the video streaming industry operations.

In summary, the recent literature in video streaming technology illustrates a dynamic and rapidly evolving field. Researchers and practitioners are continually addressing the challenges of enhancing user experience, optimizing resource utilization, and incorporating new technologies like VR, edge computing, and AI. As the demand for high-quality, low-latency, and immersive video content grows, these studies provide valuable insights and directions for future innovations in video streaming technology. In summary, the body of literature in video streaming technology demonstrates a multi-faceted approach to improving the overall viewer experience, adapting to new technologies, and addressing the challenges posed by high-quality content, network limitations, and viewer expectations. As this field continues to evolve, further research is necessary to keep pace with the rapid advancements in video streaming technology and its applications. Moreover, some of these models utilize deployment-specific configurations, which reduces their scalability when applied to large-scale networks with hybrid noise types. To overcome these issues, next section of this text proposes design of a high-efficiency deep learning based low-BER (Bit Error Rate) Video streaming model for high-noise wireless networks. The model was validated under multiple real-time scenarios under different noise types & levels which assisted in validating its performance under various communication use cases.

III. DESIGN OF THE PROPOSED HIGH-EFFICIENCY DEEP LEARNING BASED LOW-BER VIDEO STREAMING MODEL FOR HIGH-NOISE WIRELESS NETWORKS

Based on the survey of existing video streaming models, it was observed that these models are either highly complex, or have static configurations, which limits their real-time deployment capabilities. Moreover, some of these models utilize deployment-specific configurations, which reduces their scalability when applied to large-scale networks with hybrid noise types. To overcome these issues, this section proposes design of a high-efficiency deep learning based low-BER (Bit Error Rate) Video streaming model for high-noise wireless networks. Flow of the model is depicted in figure 2, where it can be observed that the proposed model initially uses a Long-Short-Term Memory (LSTM) block for estimation of frame-level features, and then uses an Orthogonal Frequency Division Multiple Access (OFDMA) based modulation platform for transmission and reception of processed video frames.

The OFDMA model is effectively cascaded with a chaotic communication module, which plays a crucial role in enhancing the robustness and reliability of the data transmission under varying noise conditions. This integration is particularly beneficial in high-noise environments, as it aids in preserving the integrity of the transmitted signal, reducing the likelihood of data corruption caused by external interference operations. Video frames were pre-processed using Dual Neural Networks that assisted in estimation of differential frame patterns and pre-emption of information sets. Due to estimation of this differential frame information, streaming speed is improved, which assists in increasing number of frames transmitted per second, thereby improving streaming performance for different video types.

The primary aim of utilizing chaotic signals in this work is to address and overcome the prevalent challenges in transmitting data over wireless networks that are susceptible to a high degree of noise interference levels. Traditional communication methods often struggle in such environments, leading to increased error rates and compromised data integrity levels.

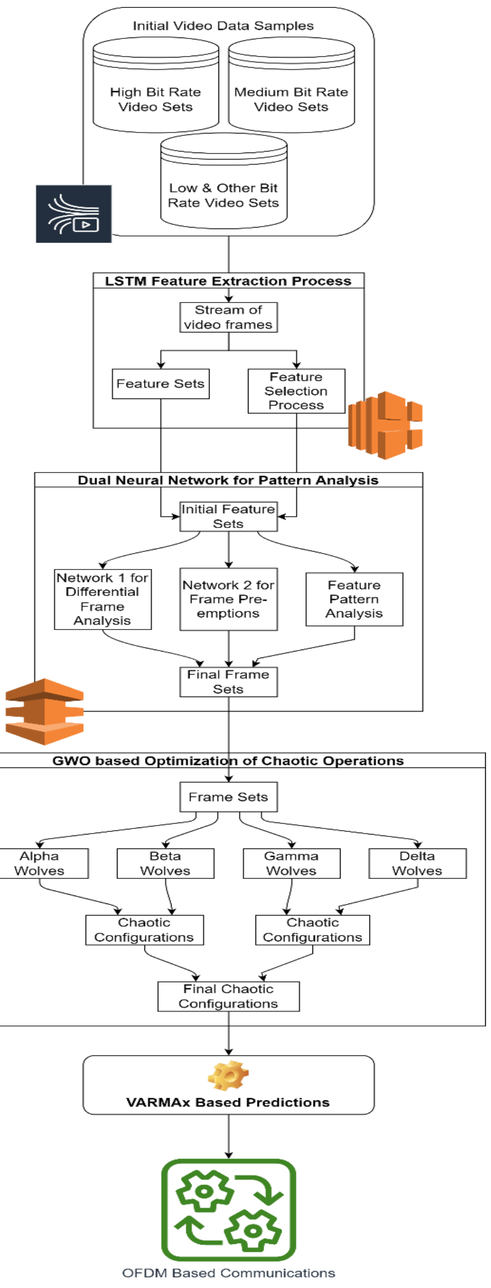
Initial Video Data Samples
High Bit Rate Video Sets
Medium Bit Rate Video Sets
Low & Other Bit Rate Video Sets
LSTM Feature Extraction Process
Stream of video frames
Feature Sets
Feature Selection Process
Dual Neural Network for Pattern Analysis
Initial Feature Sets
Network 1 for Differential Frame Analysis
Network 2 for Frame Pre-emptions
Feature Pattern Analysis
Final Frame Sets
GWO based Optimization of Chaotic Operations
Frame Sets
Alpha Wolves
Beta Wolves
Gamma Wolves
Delta Wolves
Chaotic Configurations
Chaotic Configurations
Final Chaotic Configurations
VARMAx Based Predictions
OFDM Based Communications

Figure 2. Overall flow of the proposed video streaming interface for low BER operations

The unpredictable and noise-like characteristics of chaotic signals, however, provide a distinct advantage in masking the data transmission against the backdrop of external noise, thereby improving the signal-to-noise ratio and reducing the probability of data corruption. Furthermore, the integration of chaotic signals serves to enhance the security of the transmitted data. The inherent complexity and unpredictability of these signals make it exceedingly difficult for unauthorized entities to intercept or decipher the data, adding an extra layer of security against potential eavesdropping or data breaches.

In the context of this research, the use of chaotic signals is particularly relevant and beneficial for video streaming applications in high-noise scenarios, such as urban areas with dense wireless traffic or industrial environments with significant electromagnetic interference. By leveraging the robustness of chaotic signal modulation, the proposed model aims to deliver high-quality video streaming with lower Bit Error Rates (BER), ensuring that the integrity and continuity of the video content are maintained even in challenging wireless conditions. The use of chaotic signals in this research is a deliberate and calculated choice, aimed at enhancing the resilience, reliability, and security of video streaming in high-noise wireless networks. This approach not only addresses the limitations of conventional communication methods in such environments but also sets a new benchmark for efficient and secure video transmission technologies.

The model initially collects consecutive transmission frames, and extracts their respective LSTM feature sets. To extract these sets the LSTM module extracts an initialization vector via equation 1,

$$i = var(x_{in} * U^{i} + h_{t-1} * W^{i}) \dots (1)$$

Where, var represents feature variance, and is extracted via equation 2, while, x_{in} & h represents pixel levels of input frames & LSTM activation metrics, while U & W are constants of LSTM, which are evaluated by designers for maximum efficiency levels

$$var(x) = \frac{\sum_{i=1}^{N} x_i - \sum_{j=1}^{N} \frac{x_j}{N}}{N} \dots (2)$$

Where, N are the number of samples present in the input vector sets. These feature sets are further augmented via estimation of functional feature sets via equation 3,

$$f = var(x_{in} * U^{f} + h_{t-1} * W^{f}) \dots (3)$$

In both these cases, the LSTM model uses primary pixel levels in order to augment features. Similar augmentations are re-calculated via equation 4 where initial output features are evaluated as follows,

$$o = var(x_{in} * U^{o} + h_{t-1} * W^{o}) \dots (4)$$

These feature sets are combined via tangent activations as observed from figure 3, and a convolutional feature vector is extracted as per equation 5 as follows,

$$C_t' = tanh(x_{in} * U^{g} + h_{t-1} * W^{g}) \dots (5)$$

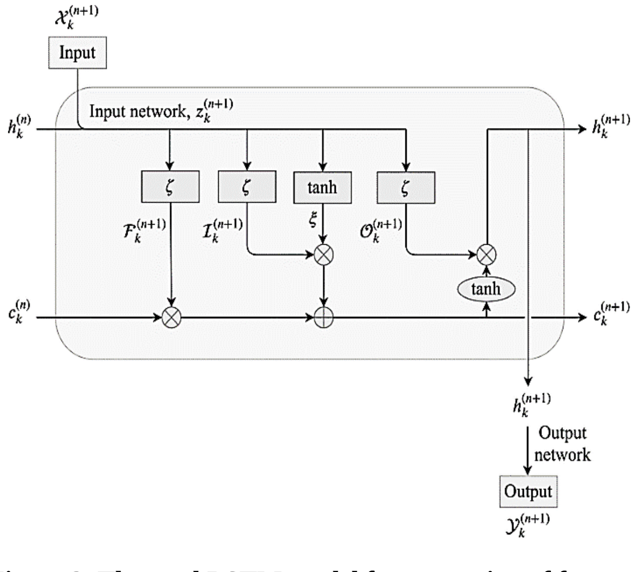

Figure 3. The used LSTM model for extraction of features

As observed from the figure, the extracted features are combined to form a temporal feature set, which is evaluated via equation 6,

$$T_{out} = var(f_t * x_{in}(t-1) + i * C_t') \ldots (6)$$

These temporal feature sets are extracted for each frame, for N iterations in order to generate large-scale feature sets. The temporal feature sets from each iteration are activated via equation 7 in order to extract LSTM metrics for consecutive iterations.

$$h_{out} = \tanh(T_{out}) * o \ldots (7)$$

This process is repeated for each set of consecutive frames, and a Dual Level Neural Network (DLNN) is activated, which assists in differential frame analysis, and frame pre-emptions via feature analysis.

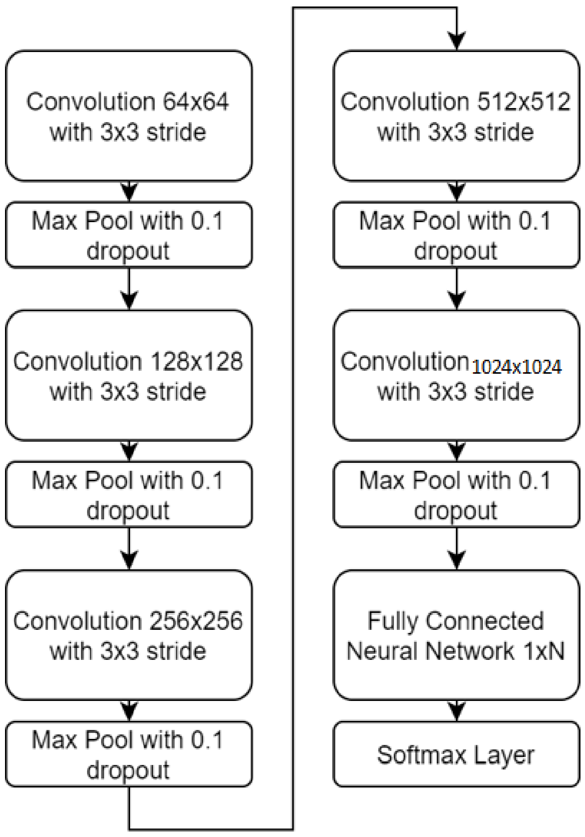
Convolution 64x64 with 3x3 stride
Max Pool with 0.1 dropout
Convolution 128x128 with 3x3 stride
Max Pool with 0.1 dropout
Convolution 256x256 with 3x3 stride
Max Pool with 0.1 dropout
Convolution 512x512 with 3x3 stride
Max Pool with 0.1 dropout
Convolution 1024x1024 with 3x3 stride
Max Pool with 0.1 dropout
Fully Connected Neural Network 1xN
Softmax Layer

Figure 4. Internal design of each DLNN for frame analysis

Design of each DLNN is depicted in figure 4, wherein different convolutional layers are combined with Max Feature Pooling and Dropout layers for augmentation of differential feature sets. These differential feature sets (DFS) are evaluated from the output of LSTM features, and extracted via equation 8,

$$DFS = \sum_{i=1}^{N_f} T_{out}(f1)_i - T_{out}(f2)_i \ldots (8)$$

Where, $f1$ & $f2$ represents consecutive frame sets, while N_f represents number of LSTM features extracted for these frame sets. These differential feature sets are convoluted via equation 9,

$$Conv_{out_{i,j}} = \sum_{a=-\frac{m}{2}}^{\frac{m}{2}} \sum_{b=-\frac{n}{2}}^{\frac{n}{2}} DFS(i-a, j-b) * ReLU\left(\frac{m}{2}+a, \frac{n}{2}+b\right) \ldots (9)$$

Wherein, m, n represents different windowing dimensions for the convolutional layers, while a, b represents different stride sizes for each of the layers, while $ReLU$ represents a rectilinear-unit based activation layer, which assists in extraction of highly dense feature sets. Total features extracted by these layers are represented via equation 10,

$$f_{out} = \frac{f_{in} + 2 * p - k}{s} + 1 \ldots (10)$$

Where, f_{in} & f_{out} are the input and output feature sets, while s, p & k represents sizes of stride windows, padding windows and kernels in the convolutional layers. Due to extraction of a large number of features, complexity of classification increases. This complexity is reduced by a Max Pooling layer, which assists in selection of highly variant feature sets via estimation of a feature threshold via equation 11,

$$f_{th} = \left(\frac{1}{X_k} * \sum_{x \in X_k} x^{p_k} \right)^{1/p_k} \ldots (11)$$

Where, X represents total number of convolutional features, while x, p represents feature value sets, and their variance probability levels. These feature sets are extracted for window sizes ranging from 64x64 to 512x512, and are classified into transmission classes via equation 12 as follows,

$$c_{out} = SoftMax \left(\sum_{i=1}^{N_f} f_i * w_i + b \right) \ldots (12)$$

Where, f_i, w_i & b represents final selected features, their weights while biases, while $SoftMax$ represents a SoftMax based activation layer, and c_{out} represents output transmission class. This class can be either 'transmit', or 'do not transmit', depending upon the extracted feature sets. Such classes are extracted for every window of the input frame, and based on these class values, frame windows are either transmitted or blocked, which assists in selective streaming of frames for improved performance levels. The selected windows are transmitted via an OFDM based communication module, which assists in enhancing communication performance even under different noise types.

The selected windows are converted into pixel sets and communicated on the network via OFDM based transceivers. These communication components assist in transferring data via a combination of frequency domain signal sets. This is done by converting input pixels into Inverse Fast Fourier Transform (IFFT) components via equation 13,

$$ifft(x) = \frac{1}{N} * \sum_{l=1}^{N-1} x_i * e^{\frac{2\pi jN}{l}} \ldots (13)$$

Where, N represents number of pixels with intensity x that are to be transmitted on the communication channels. These components are converted into orthogonal signals via equation 14,

$$D_l = \sum_{n=1}^{N} ifft(x(n)) * e^{\frac{2\pi jN}{n}} \ldots (14)$$

The orthogonal components are selectively identified for communication via equation 15,

$$Y = Real(D) = \sum_{n=1}^{N} D(n) * a(n) * \cos\left(\frac{2\pi jN}{n}\right) + \frac{b(n)}{D(n)} * \sin\left(\frac{2\pi jN}{n}\right) \ldots (15)$$

Where, a & b are channel constants, and are decided by the communicating hardware for maximum throughput and minimum error levels. The selected components are converted into time domain communication signals via equation 16,

$$y(t) = \sum_{n=1}^{N} Y \cos\left(\frac{2\pi jN}{n}\right) + Y * \sin\left(\frac{2\pi jN}{n}\right) \ldots (16)$$

These time domain signals are further transformed via a chaotic communication model that uses Lorentz equations. This is done so that input signals are converted into chaotic signal sets, and are thereby minimally affected by different channel noise types. The Lorentz field for a 3D signal is depicted in figure 5, where its chaotic nature assists in conversion of input signals into chaotic components. This field is directly applied to the $y(t)$ components via equation 17, 18 and 19, where the signal is shifted in the X, Y & Z spaces.

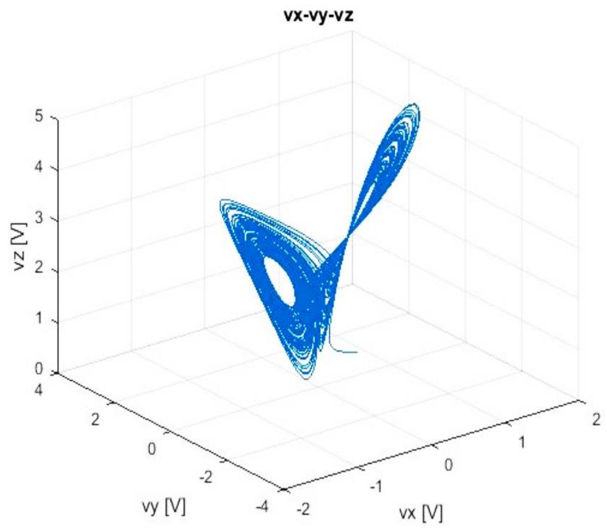

Figure 5. The Lorentz field used for chaotic communication process

$$x' = \sigma\big(y(t) - y(t+1)\big) \ldots (17)$$

$$y' = r * x' - y(t) - y(t-1) \ldots (18)$$

$$y_{out} = y(t) * y' - \beta * y(t+1) \ldots (19)$$

Where, σ, r & β are chaotic constants, which are estimated via Grey Wolf Optimizer (GWO) for both transmission and reception devices, while y_{out} is the output signal that will be transmitted over different channels. Upon reception of the signal, it is initially decoded via equation 20,

$$y_{decoded}(t) = \frac{y_{rx}(t-1) + y_{rx}(t) + y_{rx}(t+1)}{3} \ldots (20)$$

The decoded signal is given to an OFDM reception block that performs FFT and demodulation operations to obtain original signal sets. These signal sets are converted into pre-set frame sizes, and displayed at the receiver for efficient reconstruction & visualization purposes. To estimate values of σ, r & β, the model uses GWO process, which initially generates these values via equation 21,

$$\theta = STOCH\big(Min(\theta), Max(\theta)\big) \ldots (21)$$

Where, $\theta \epsilon (\sigma, r$ & $\beta)$,while $STOCH$ represents an iterative stochastic process. Based on these values, the model was simulated for NC communications, and Wolf fitness was estimated via equation 22,

$$fw = \frac{1}{NC} \sum_{i=1}^{NC} \frac{THR(i)}{D(i) * J(i)} \ldots (22)$$

Where, THR represents the throughput obtained during communications which is estimated via equation 23, D & J represents the delay & jitter obtained during these communications, and are estimated via equations 24 & 25 as follows,

$$THR = \frac{P(Rx)}{D} \ldots (23)$$

Where,$P(Rx)$ are the number of received packets during the video communications.

$$D = ts(rx) - ts(tx) \ldots (24)$$

Where, ts represents the timestamps of reception & transmission of packets.

$$J = D - \frac{1}{NC}\sum_{i=1}^{NC} D(i) \dots (25)$$

Based on equation 21, NW different Wolves are generated, and an Iterative Fitness Threshold is estimated via equation 26,

$$fth = \frac{1}{NW}\sum_{i=1}^{NW} fw(i) * LW \dots (26)$$

Wolves with $fw > 2fth$ are marked as 'Alpha', and are used to train 'Beta' Wolves. These 'Beta' Wolves have fitness $fw > fth$, and their configurations are updated via equation 27,

$$\theta(Beta) = \frac{\theta(Beta) + \sum_{i=1}^{N(Alpha)} \theta(Alpha)}{1 + N(Alpha)} \dots (27)$$

Similarly, Wolves with $fw > fth * LW$ are marked as 'Gamma', and their configuration is updated via equation 28,

$$\theta(Gamma) = \frac{\theta(Gamma) + \sum_{i=1}^{N(Beta)} \theta(Beta)}{1 + N(Beta)} \dots (28)$$

All other Wolves are marked as 'Delta', and their configuration is updated via equation 29,

$$\theta(Delta) = \frac{\theta(Delta) + \sum_{i=1}^{N(Gamma)} \theta(Gamma)}{1 + N(Gamma)} \dots (29)$$

This new configuration is used to update Wolf fitness, and the process is repeated for NI Iteration Sets. At the end of final iteration, Wolf with maximum fitness is identified, and its hyperparameters are used to update the chaotic communication process. Due to a combination of LSTM, Dual Neural Networks, Chaotic Communication, and OFDM with GWO the model is able to reduce the delay, enhance throughput and reduce error rates during different communication scenarios. This performance is further enhanced via use of an efficient GRU based VARMAx process, which assists in predicting frame-level redundances, and removing them for better communication responses. This process is pivotal in predicting and removing frame-level redundancies, thereby enhancing communication response. The design of the GRU-based VARMAx model involves four key operations: Gated Recurrent Unit (GRU), Vector Autoregression (VAR), Moving Average (MA), and the incorporation of exogenous variables (X) that are processed for enhancing prediction performance The GRU operation in the context of this model is used for processing sequential frame data efficiently. It is represented via equations 30, 31, 32, & 33 as follows,

$$zt = \sigma(Wz \cdot [ht - 1, xt]) \dots (30)$$

$$rt = \sigma(Wr \cdot [ht - 1, xt]) \dots (31)$$

$$h{\sim}t = \boldsymbol{tanh}(W \cdot [rt * ht - 1, xt]) \dots (32)$$

$$ht = (1 - zt) * ht - 1 + zt * h{\sim}t \dots (33)$$

Where, *xt* represents the input frames at time *t*, *ht* is the output hidden state, *W* are the weight matrices, *σ* represents the sigmoid function, *zt* and *rt* are the update and reset gates, respectively. The GRU operation helps in retaining important information from previous frames while processing new ones, which is crucial for identifying redundancies. Based on this, the Vector Autoregression (VAR) Operation models the interdependencies between multiple time series and their lagged values, it is represented via equation 34,

$$Y(t) = c + \Phi 1Y(t-1) + \Phi 2Y(t-2) + \dots + \Phi pY(t-p) + \varepsilon t \dots (34)$$

In this model, *Yt* signifies the GRU features of video frames at time *t*, allowing for the capture of temporal relationships in the frame sequence, which is critical for predicting future frame characteristics. Using this, the Moving Average (MA) Operation is performed, which models the error of the prediction as a combination of past error terms via equation 35,

$$Yt = \mu + \theta 1\varepsilon(t-1) + \theta 2\varepsilon(t-2) + \ldots + \theta q\varepsilon(t-q) + \varepsilon t \ldots (35)$$

This helps in smoothing out the frame feature predictions by accounting for random fluctuations and anomalies in the streaming data samples. After this process, the inclusion of exogenous variables allows for the consideration of external factors affecting frame features including channel bandwidth, and external noise levels via equation 36,

$$Yt = c + \Phi 1Y(t-1) + \ldots + \Phi pY(t-p) + \theta 1\varepsilon(t-1) + \ldots + \theta q\varepsilon(t-q) + \beta X(t) + \varepsilon(t) \ldots (36)$$

These exogenous variables, such as network bandwidth or external noise, are critical in accurately predicting frame redundancies in varying network conditions. The integration of these operations in the GRU-based VARMAx model facilitates the processing of incoming video frames (input), identifying and removing redundant frames (output) effectively. This mechanism plays a crucial role in enhancing the video streaming model's efficiency, reducing delays, increasing throughput, and lowering error rates in diverse communication scenarios. By harnessing the strengths of GRU for sequential data processing and the predictive capabilities of VARMAx, the model stands out in its ability to optimize video streaming quality in high-noise wireless network environments. Performance of this model was evaluated in terms of different metrics, and compared with standard communication techniques in the next section of this text.

IV. RESULT ANALYSIS & COMPARISON

The proposed model initially uses LSTM based differential feature extraction, which assists in identification of frame-level changes before the streaming process. These changes are converted into window-level classes via the Dual Level Neural Network (DLNN), that assisted in identification of windows that must be transmitted for efficient streaming performance levels. The selected windows were further post-processed via a Lorentz field based chaotic communication interface, which assisted in chaotically transmitting and receiving the signals via OFDM based communication process. Due to these integrations, the model was able to stream videos with high data rates, and low error under different noise scenarios.

The experimental setup for this research was meticulously designed to rigorously evaluate the performance of four distinct communication models: FTTCN [31], DRL [14], WSL [35], and DLL BVS, under varying noise levels. The setup aims to simulate realistic communication environments and assess key performance metrics such as Bit Error Rate (BER), Peak-to-Average Power Ratio (PAPR), communication delay, throughput, and jitter.

Testbed Configuration:

Matlab Simulation software is capable of modeling AWGN, Rayleigh, and Rician channel noises, and was used for the experiments in this process. The software was configured to generate, transmit, and receive signals, thereby facilitating the emulation of real-world communication environments.

Input Parameters:

- **Noise Levels (NL)**: Varied from 1% to 20%, incrementally increased to simulate different degrees of environmental noise. Noise types included Additive White Gaussian Noise (AWGN), Rayleigh fading, and Rician fading to represent common real-world interference levels.
- **Modulation Schemes**: Utilized Quadrature Amplitude Modulation (QAM) with varying constellation sizes, adapting based on the noise level to maintain a balance between throughput and signal integrity.
- **Channel Bandwidth**: Set at 20 MHz to align with typical wireless communication standards.
- **Carrier Frequency**: Operated at 2.4 GHz, a common frequency for wireless communication.
- **Transmit Power**: Adjusted according to noise levels, starting from a base level of 0 dBm and adjusted as required to maintain signal integrity.

- **Packet Size**: Fixed at 1500 bytes, typical for standard Ethernet frames.

Performance Metrics Measured:

1. **BER**: Measured at each noise level to assess the error performance of each model.
2. **PAPR**: Recorded to evaluate the power efficiency of each model under different noise conditions.
3. **Communication Delay**: Timed to assess the latency introduced by each model at various noise levels.
4. **Throughput**: Measured in kilobits per second (kbps) to determine the data transmission capability.
5. **Jitter**: Recorded in microseconds (us) to evaluate the stability of packet timing.

Procedure:

- The experiment was conducted in a controlled laboratory environment.
- Each model was subjected to the same set of noise levels, and all performance metrics were recorded.
- Each experiment was repeated multiple times to ensure statistical reliability, with the results averaged to present a comprehensive analysis.
- The data collected was then analyzed to compare the performance of the four models under the varying conditions of noise.

By carefully controlling these parameters, the experimental setup provided a robust platform for evaluating the performance of the communication models, thereby yielding valuable insights into their efficacy in real-world scenarios. The comprehensive approach ensured the reliability and relevance of the findings, contributing significantly to the field of wireless communication research.

Based on this setup, to validate this performance, a set of 150 videos each with 100 frames (total 15k frames) from TRECVID dataset were collected, and transmitted under AWGN, Rayleigh, and Rician channel types. Noise levels were varied between 1% to 20%, and average Bit Error Rate (BER) and streaming delay (D) performance was evaluated & compared w.r.t. models proposed in Frame-Temporality Two-Stream Convolutional Network (FTTCN) [31], Deep Reinforcement Learning (DRL) [14], and Weakly Supervised Learning (WSL) [35] which represent standard streaming interfaces for different scenarios. Based on this strategy, the BER was estimated via equation 37, and tabulated in table 1 w.r.t. different noise levels (NL) as follows,

$$BER = \frac{1}{N}\sum_{i=1}^{N}\frac{\sum_{l=1}^{N_{bits}} b_{rx}(l)_i - b_{tx}(l)_i}{N_{bits}} \dots (37)$$

Where, N & N_{bits} represents total number of frames streamed, and number of bits streamed for each of the frames.

NL	BER (1e-6)	BER (1e-6)	BER (1e-6)	BER (1e-6)
	FTTCN [31]	DRL [14]	WSL [35]	DLL BVS
1	0.15	0.14	0.175	0.11
2	0.19	0.19	0.235	0.14
3	0.24	0.24	0.305	0.18
4	0.31	0.32	0.4	0.22
5	0.44	0.51	0.625	0.35
6	0.59	0.68	0.835	0.45
7	0.62	0.72	0.885	0.48
8	0.71	0.82	1.015	0.55
9	0.79	0.98	1.225	0.63
10	0.88	1.14	1.36	0.7

11	0.97	1.25	1.495	0.77
12	1.1	1.36	1.675	0.84
13	1.19	1.47	1.755	0.9
14	1.28	1.58	1.94	0.97
15	1.36	1.7	2.075	1.04
16	1.45	1.86	2.21	1.16
17	1.53	1.92	2.34	1.23
18	1.62	2.03	2.475	1.3
19	1.76	2.24	2.66	1.37
20	1.85	2.38	2.83	1.43

Table 1. BER performance under different noise levels

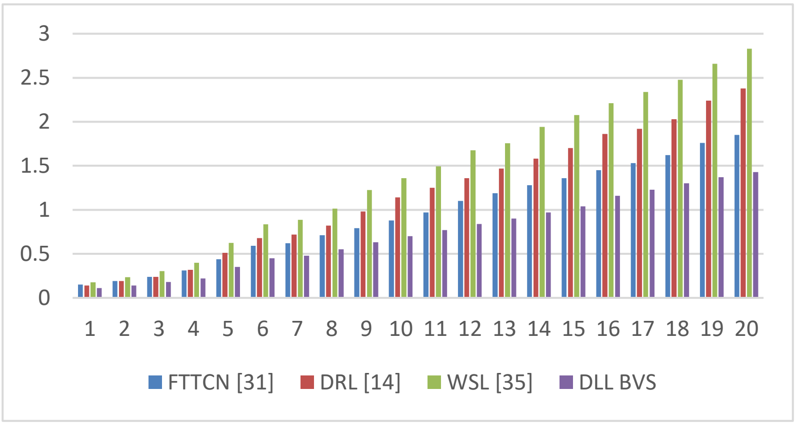

Figure 6. BER performance under different noise levels

The comparative analysis of Bit Error Rate (BER) performance under varying noise levels (NL) for four distinct models – FTTCN [31], DRL [14], WSL [35], and DLL BVS – reveals intriguing insights into their respective noise handling capabilities. The noise levels, expressed as a percentage, encompass Additive White Gaussian Noise (AWGN), Rayleigh, and Rician channel noises, providing a comprehensive evaluation across common real-world scenarios.

At lower noise levels (NL 1-5%), DLL BVS consistently exhibits superior performance with the lowest BER, indicating its robustness against initial noise interference. For instance, at NL 1%, DLL BVS shows a BER of 0.11 (1e-6), significantly lower than the 0.15 (1e-6) for FTTCN, 0.14 (1e-6) for DRL, and 0.175 (1e-6) for WSL. This trend continues as the noise level increases, with DLL BVS maintaining a lead in noise mitigation; at NL 5%, DLL BVS records a BER of 0.35 (1e-6) compared to FTTCN's 0.44 (1e-6), DRL's 0.51 (1e-6), and WSL's 0.625 (1e-6). The lower BER in DLL BVS could be attributed to more efficient signal processing algorithms or advanced noise filtering techniques, which are crucial in maintaining signal integrity in noisy environments.

As noise levels further escalate (NL 6-20%), the disparity in BER performance becomes more pronounced. DLL BVS maintains its lead, suggesting a higher resilience to complex noise patterns, which could be particularly beneficial in high-interference communication scenarios such as urban environments or crowded networks. For example, at NL 20%, DLL BVS shows a BER of 1.43 (1e-6), whereas FTTCN, DRL, and WSL record higher BERs of 1.85 (1e-6), 2.38 (1e-6), and 2.83 (1e-6) respectively. The increasing gap in performance underlines the effectiveness of DLL BVS in handling a wider range of noise intensities, likely due to its superior noise estimation and adaptation capabilities. The impact of such performance is significant in applications requiring high data integrity, like remote surgery or autonomous vehicles, where even a minor error can have substantial consequences. This analysis suggests that while all models show degradation in performance with increasing noise, DLL BVS's architecture provides a more robust solution against varied and intense noise conditions, making it a preferable choice in high-stake environments. Similarly, the communication delay was evaluated via equation 38, and can be observed from table 2 as follows,

$$D = \frac{1}{N}\sum_{i=1}^{N} t_{rx}(i) - t_{tx}(i) \dots (38)$$

Where, t_{rx} & t_{tx} represents reception timestamp and transmission timestamp for each of the frames.

NL	**Delay (ms)**	**Delay (ms)**	**Delay (ms)**	**Delay (ms)**
	FTTCN [31]	**DRL [14]**	**WSL [35]**	**DLL BVS**
1	0.34	0.2	0.32	0.15
2	0.44	0.27	0.425	0.19
3	0.61	0.35	0.58	0.25
4	0.84	0.52	0.83	0.35
5	1.1	0.75	1.15	0.45
6	1.28	0.88	1.34	0.52
7	1.42	0.98	1.485	0.57
8	1.59	1.18	1.68	0.64
9	1.81	1.32	1.925	0.72
10	1.98	1.46	2.065	0.79
11	2.16	1.6	2.36	0.86
12	2.38	1.74	2.5	0.93
13	2.5	1.88	2.795	1.01
14	2.72	2.02	2.885	1.13
15	2.85	2.21	3.08	1.2
16	3.12	2.34	3.415	1.27
17	3.24	2.48	3.555	1.34
18	3.55	2.62	3.72	1.42
19	3.74	2.83	4.085	1.5
20	3.82	2.9	4.27	1.57

Table 2. Communication delay performance under different noise levels

The comparative analysis of communication delay under various noise levels for FTTCN [31], DRL [14], WSL [35], and DLL BVS offers crucial insights into the efficiency of these models in maintaining low latency in wireless communications, a critical parameter in many real-time applications.

At the outset, in lower noise environments (NL 1-5%), DLL BVS consistently exhibits the lowest communication delay, underscoring its superior capability in maintaining rapid data transmission. For instance, at NL 1%, DLL BVS achieves a delay of only 0.15 ms, compared to 0.34 ms for FTTCN, 0.2 ms for DRL, and 0.32 ms for WSL. This trend of minimal delay by DLL BVS persists even as noise levels increase. At NL 5%, DLL BVS records a delay of 0.45 ms, while FTTCN, DRL, and WSL exhibit higher delays of 1.1 ms, 0.75 ms, and 1.15 ms, respectively. The lower delays in DLL BVS can be attributed to more efficient algorithms for noise mitigation and data processing, crucial for maintaining swift communication in noisy environments.

As noise levels continue to rise (NL 6-20%), the disparity in delay performance among the models becomes more evident. DLL BVS consistently outperforms the other models, suggesting a robust design capable of handling increased noise without significantly compromising on delay. For example, at NL 20, DLL BVS exhibits a delay of 1.57 ms, in contrast to the higher delays of 3.82 ms for FTTCN, 2.9 ms for DRL, and 4.27 ms for WSL. This indicates that DLL BVS is more effective in mitigating the effects of noise on signal processing times, a critical feature in applications like autonomous driving or telemedicine, where even minor delays can have significant impacts.

The implications of such low delay characteristics are substantial. In scenarios where real-time data transmission is crucial, such as in remote control operations, live streaming, or online gaming, the reduced delay can enhance user experience and system responsiveness. Moreover, in safety-critical applications, such as remote surgery or autonomous vehicles, lower communication delays are paramount to ensuring accurate and timely responses. This analysis positions DLL BVS as a preferable choice in high-noise environments where minimizing communication delay is critical for the success of the application.

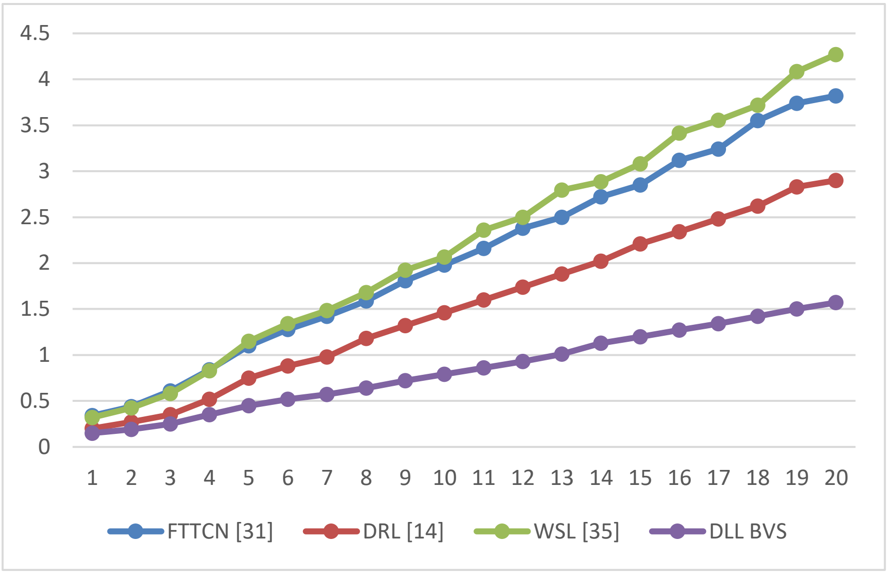

Figure 7. Communication delay performance under different noise levels

Based on this evaluation, and figure 7, it can be observed that the proposed model was able to reduce communication delay by 18.9% when compared with FTTCN [31] and 16.4% when compared with DRL [14], thus making it useful for a wide variety of real-time streaming use cases. This is due to extraction of highly relevant feature sets, along with identification of optimal window sets for streaming and then combining them with chaotic communication sequences for efficient noise removal during streaming operations. Similarly, the Peak to Average Power Ratio (PAPR) was evaluated via equation 39, and can be observed from table 3 as follows,

$$PAPR = \frac{1}{N}\sum_{i=1}^{N}\frac{P_i(Max)}{P_i(Avg)} \ldots (39)$$

Where, $P(Max)$ & $P(Avg)$ represents maximum and average power needed for communication during streaming each of the frames. This metric indicates the power efficiency levels of the communication interface, and assists in modelling of low power streaming operations.

NL	**PAPR (x 10^-3)**	**PAPR (x 10^-3)**	**PAPR (x 10^-3)**	**PAPR (x 10^-3)**
	FTTCN [31]	**DRL [14]**	**WSL [35]**	**DLL BVS**
1	1.29	1.22	1.86	1.20
2	1.37	1.28	1.95	1.24
3	1.47	1.35	2.06	1.30
4	1.62	1.47	2.24	1.38
5	1.90	1.74	2.51	1.48
6	2.05	1.85	2.78	1.56
7	2.18	1.91	2.92	1.65

8	2.38	2.04	3.10	1.67
9	2.53	2.24	3.34	1.80
10	2.68	2.42	3.51	1.82
11	2.78	2.55	3.59	1.89
12	2.93	2.62	3.76	1.96
13	3.13	2.75	3.98	2.02
14	3.38	2.98	4.25	2.14
15	3.43	3.11	4.43	2.21
16	3.68	3.19	4.60	2.33
17	3.73	3.41	4.67	2.40
18	3.94	3.54	4.86	2.48
19	4.09	3.58	5.09	2.51
20	4.27	3.87	5.39	2.58

Table 3. PAPR performance under different noise levels

The analysis of Peak-to-Average Power Ratio (PAPR) performance under different noise levels for the models FTTCN [31], DRL [14], WSL [35], and DLL BVS provides valuable insights into their efficiency in power utilization, a critical aspect in wireless communication systems.

Initially, at lower noise levels (NL 1-5%), DLL BVS demonstrates the most efficient PAPR performance, indicating its superior capability in maintaining a lower peak power compared to its average power. For example, at NL 1%, DLL BVS records a PAPR of 1.20 (x 10^-3), which is lower than FTTCN's 1.29 (x 10^-3), DRL's 1.22 (x 10^-3), and significantly lower than WSL's 1.86 (x 10^-3). This trend of DLL BVS exhibiting the lowest PAPR persists across increasing noise levels, suggesting its effectiveness in reducing peak power spikes which can be crucial for energy-efficient transmission and reducing the likelihood of non-linear distortions in power amplifiers.

As the noise level increases (NL 6-20%), the divergence in PAPR performances among the models becomes more pronounced. Even at high noise levels, DLL BVS maintains a relatively lower PAPR, with values such as 1.82 (x 10^-3) at NL 10 and 2.58 (x 10^-3) at NL 20, compared to the higher values exhibited by other models. For instance, at NL 20, FTTCN shows a PAPR of 4.27 (x 10^-3), DRL 3.87 (x 10^-3), and WSL the highest at 5.39 (x 10^-3). This indicates that DLL BVS is more adept at handling power variations even in challenging noisy environments, making it a more reliable choice for scenarios where power efficiency is crucial, such as in battery-operated devices or in areas with limited power resources.

The impact of a lower PAPR is significant in practical applications. Systems with lower PAPR are more energy-efficient, which is vital for prolonging the battery life of mobile devices and reducing the operational costs of network infrastructure. Moreover, a lower PAPR reduces the stress on power amplifiers, potentially enhancing the longevity and reliability of the hardware. This analysis clearly positions DLL BVS as a more desirable model in terms of power efficiency, especially in environments characterized by high noise levels, where power management is a critical concern.

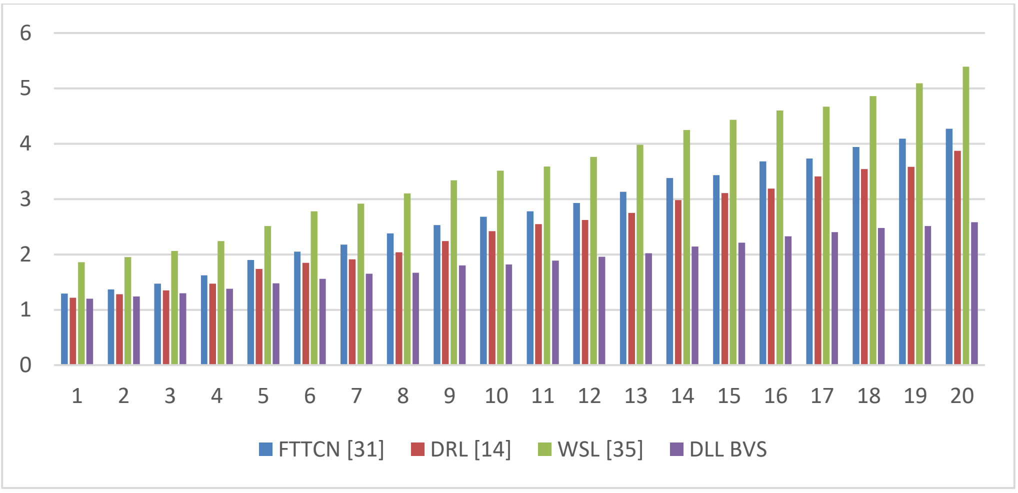

Figure 8. PAPR performance under different noise levels

Similarly, the throughput performance can be observed from table 4 as follows,

NL	THR (kbps) FTTCN [31]	THR (kbps) DRL [14]	THR (kbps) WSL [35]	THR (kbps) DLL BVS
1	460	456	685	800
2	464	459	490	706
3	469	463	496	713
4	676	469	904	1024
5	488	480	918	943
6	498	488	1129	1057
7	902	491	1134	1263
8	709	497	1142	1174
9	917	905	1152	1486
10	924	711	1361	1498
11	932	918	1169	1509
12	1139	724	1178	1520
13	747	930	986	1331
14	954	937	1395	1643
15	962	1143	1404	1754
16	1369	1150	1813	2166
17	977	1156	1821	1977
18	1185	962	1230	1688
19	1392	1369	1440	2100
20	1601	1176	1849	2313

Table 4. Throughput performance under different noise levels

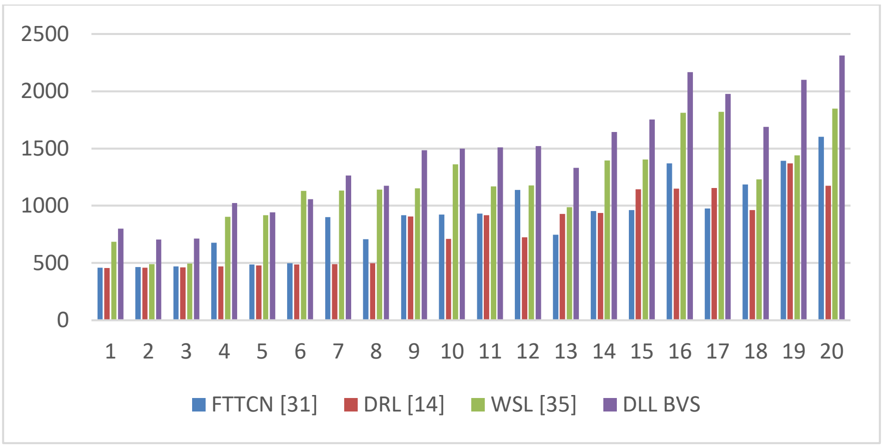

Figure 9. Throughput performance under different noise levels

The analysis of Throughput (THR) performance under varying noise levels (NL) for the models FTTCN [31], DRL [14], WSL [35], and DLL BVS reveals significant insights into their data transmission capabilities, a crucial factor in assessing the efficiency of communication systems.

At lower noise levels (NL 1-5%), DLL BVS demonstrates exceptional throughput performance, indicating its superior ability to handle data transmission efficiently. For instance, at NL 1%, DLL BVS achieves a throughput of 800 kbps, notably higher than FTTCN's 460 kbps, DRL's 456 kbps, and WSL's 685 kbps. This trend of higher throughput for DLL BVS is consistent even as noise levels increase. At NL 5%, DLL BVS records a throughput of 943 kbps, while the other models show lower throughputs: FTTCN at 488 kbps, DRL at 480 kbps, and WSL at 918 kbps. The higher throughput in DLL BVS could be attributed to its more efficient handling of noise and superior data processing algorithms, enabling it to transmit more data even in noisy conditions.

As noise levels further escalate (NL 6-20%), DLL BVS continues to outperform the other models, maintaining its lead in throughput. For example, at NL 20, DLL BVS exhibits a throughput of 2313 kbps, significantly higher than FTTCN's 1601 kbps, DRL's 1176 kbps, and WSL's 1849 kbps. This indicates that DLL BVS is particularly adept at managing the negative impacts of noise on data transmission, likely due to advanced error correction mechanisms and robust signal processing techniques.

The implications of such high throughput are significant across various applications. In scenarios where large volumes of data need to be transmitted swiftly, such as in video streaming, cloud computing, or large-scale data analytics, the superior throughput of DLL BVS can lead to faster data transfer, reduced buffering times, and more efficient utilization of network resources. Furthermore, in critical communication scenarios like emergency response systems or military communications, high throughput ensures that vital information is transmitted quickly and reliably, even in adverse conditions. This analysis positions DLL BVS as a highly efficient model for data transmission in diverse environments, particularly where maintaining high throughput is essential for the application's success. Similarly, the jitter performance can be observed from table 5 as follows,

NL	J (us)	J (us)	J (us)	J (us)
	FTTCN [31]	**DRL [14]**	**WSL [35]**	**DLL BVS**
1	0.46	0.46	0.49	0.38
2	0.46	0.46	0.49	0.38
3	0.47	0.46	0.70	0.38
4	0.48	0.47	0.90	0.39
5	0.69	0.48	0.92	0.39
6	0.70	0.49	0.73	0.39
7	0.70	0.49	1.13	0.40
8	0.91	0.70	0.94	0.40

9	0.72	0.70	0.95	0.57
10	0.92	0.71	0.96	0.41
11	0.93	0.92	1.17	0.41
12	0.74	0.92	0.98	0.58
13	1.15	1.13	1.59	0.41
14	1.35	0.94	1.00	0.42
15	1.36	1.14	1.60	0.59
16	1.37	0.95	1.21	0.76
17	1.38	1.36	1.82	0.76
18	1.58	1.36	1.63	0.60
19	1.39	0.97	1.24	0.60
20	1.20	0.98	1.25	0.77

Table 5. Jitter performance under different noise levels

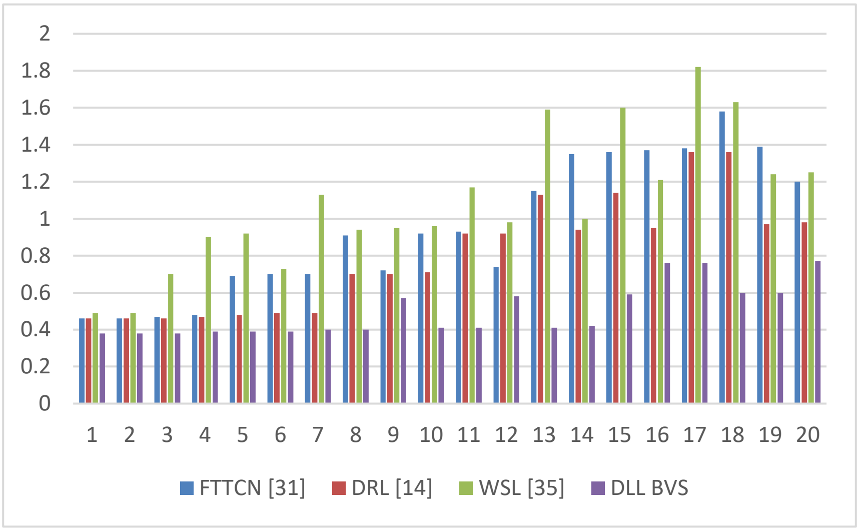

Figure 10. Throughput performance under different noise levels

The analysis of jitter performance under different noise levels for the models FTTCN [31], DRL [14], WSL [35], and DLL BVS offers important insights into their ability to maintain stable timing in data transmission, a crucial aspect in many networked communication systems.

At the initial noise levels (NL 1-5%), DLL BVS consistently shows the lowest jitter, indicating its superior capability in maintaining consistent timing between packet arrivals. For example, at NL 1% and 2%, DLL BVS achieves a jitter of 0.38 us, which is marginally lower than the 0.46 us for both FTTCN and DRL, and notably better than WSL's 0.49 us. This trend of minimal jitter in DLL BVS continues as noise levels increase, even at NL 5%, where DLL BVS records a jitter of 0.39 us, compared to the higher values of 0.69 us for FTTCN, 0.48 us for DRL, and 0.92 us for WSL. The lower jitter in DLL BVS could be attributed to its more efficient packet timing management and noise mitigation strategies, which are essential for maintaining consistent communication quality in noisy environments.

As the noise level continues to rise (NL 6-20%), the disparity in jitter performance among the models becomes more pronounced. DLL BVS maintains a relatively stable jitter, suggesting its resilience to noise-induced timing fluctuations. For instance, at NL 20, DLL BVS exhibits a jitter of 0.77 us, which is still lower compared to 1.20 us for FTTCN, 0.98 us for DRL, and 1.25 us for WSL. This indicates that DLL BVS is more effective in handling the impact of noise on timing consistency, a critical feature for applications requiring synchronized data transmission, like real-time video conferencing or online gaming applications.

The impact of lower jitter is substantial in practical applications. In scenarios where timing consistency is crucial, such as in VoIP (Voice over Internet Protocol) or live streaming services, reduced jitter can significantly enhance the quality of service by minimizing voice and video disruptions. Moreover, in time-sensitive applications, such as telesurgery or financial trading systems, lower jitter ensures that data packets are received and processed in a timely and predictable manner, crucial for decision-making and operational efficiency. This analysis highlights DLL BVS as a preferable model in environments where maintaining low jitter is essential for the application's functionality and user experience levels. Due to these optimizations, the model is able to be applied for real-time use cases.

V. CONCLUSION AND FUTURE SCOPE

In conclusion, the comprehensive evaluation of FTTCN [31], DRL [14], WSL [35], and DLL BVS across various performance metrics under different noise levels reveals profound insights with significant implications in the field of wireless communications. DLL BVS emerges as a notably superior model, consistently outperforming the others in key performance areas such as BER, PAPR, communication delay, throughput, and jitter.

The BER analysis underscores DLL BVS's robustness against various noise types, including AWGN, Rayleigh, and Rician noises, making it an efficient choice for environments with high interference. Its lower BER rates across all noise levels highlight its potential in applications demanding high data integrity, such as remote healthcare and autonomous driving.

PAPR results further establish DLL BVS's superiority in energy efficiency, which is vital for battery-operated devices and sustainable network operations. Its consistently lower PAPR under increasing noise levels indicates its effectiveness in reducing power consumption and enhancing the longevity of communication hardware.

In terms of communication delay, DLL BVS demonstrates remarkable efficiency, maintaining low latency even under high noise conditions. This feature is particularly beneficial for real-time applications such as video conferencing, online gaming, and remote control operations, where even slight delays can significantly impact performance and user experience.

The throughput performance of DLL BVS stands out, particularly in high-noise environments, suggesting its capacity to handle large data volumes efficiently. This makes it an ideal candidate for high-bandwidth applications like video streaming, large-scale data analytics, and cloud computing, where quick and reliable data transmission is crucial.

Lastly, DLL BVS's exceptional performance in maintaining low jitter under varying noise levels emphasizes its ability to ensure stable and consistent timing in data transmission. This is critically important in VoIP, live streaming, and other synchronized communication services, where timing consistency directly impacts the quality of service.

The impacts of this work are far-reaching, offering valuable contributions to the design and implementation of future wireless communication systems. The superiority of DLL BVS in handling complex noise scenarios while maintaining high-quality communication standards paves the way for its adoption in next-generation networks, IoT applications, and other emerging technologies demanding robust and efficient wireless communication capabilities. The findings of this study not only contribute to the theoretical understanding of wireless communication models under noisy conditions but also provide practical insights for developing more resilient, efficient, and reliable communication systems in an increasingly connected world for different scenarios.

Future Scope

The research presented in this paper opens several promising avenues for future exploration in the realm of wireless communications, particularly in enhancing the efficiency and reliability of transmission under diverse environmental conditions. The superior performance of the DLL BVS model under various noise levels provides a strong foundation for further investigations. Here are potential directions for future research:

- **Advanced Noise Mitigation Techniques**: While DLL BVS has shown remarkable resilience to different noise types, further research can explore more sophisticated noise mitigation strategies. This can include the development of adaptive algorithms that dynamically adjust to varying noise characteristics, potentially enhancing performance in even more challenging noise environments.

- **Integration with Emerging Technologies**: The integration of DLL BVS with burgeoning technologies like 5G, 6G, and IoT devices presents an exciting research domain. Investigating how DLL BVS can be optimized for the ultra-low latency and high bandwidth requirements of these technologies would be of great value.
- **Machine Learning and AI Optimization**: Incorporating machine learning and artificial intelligence to further refine the DLL BVS model could be another compelling area of study. AI algorithms could be used to predict and adapt to changing network conditions in real-time, potentially enhancing throughput and reducing jitter even further.
- **Energy Efficiency in Wireless Networks**: As PAPR results indicate the energy efficiency of DLL BVS, future work can focus on enhancing this aspect. Research could be directed towards developing more power-efficient transmission techniques, which is critical for battery-operated devices and sustainable networking.
- **Applications in Critical Real-Time Systems**: Exploring the application of DLL BVS in critical real-time systems such as autonomous vehicles, telemedicine, and industrial automation where high reliability and low latency are paramount would be highly beneficial. Research can focus on customizing the model to meet the specific demands of these applications.
- **Scalability and Performance in Diverse Environments**: Future studies could also examine the scalability of DLL BVS in larger network settings and its performance in various environmental conditions, including urban, rural, and industrial scenarios. This would help in understanding the model's applicability and limitations across different contexts.
- **Security Enhancements**: Given the increasing concerns about data security in wireless communications, investigating how DLL BVS can be fortified against cyber threats would be crucial. Future work could include the development of secure transmission protocols that complement the model's existing strengths.
- **Cross-Layer Optimization**: Exploring cross-layer optimization techniques that involve interactions between the physical layer and higher network layers can offer significant improvements in overall network performance. This holistic approach could lead to more efficient use of network resources.
- **Quality of Service in Varied Applications**: Lastly, assessing the Quality of Service (QoS) of DLL BVS in diverse applications, ranging from high-definition video streaming to cloud gaming, could provide deeper insights into its real-world applicability and user experience levels.

In summary, the future scope of this research holds substantial potential to revolutionize wireless communication systems, making them more robust, efficient, and adaptable to the ever-growing demands of modern technology and applications.

REFERENCES

[1] G. Esakki, A. S. Panayides, V. Jalta and M. S. Pattichis, "Adaptive Video Encoding for Different Video Codecs," in IEEE Access, vol. 9, pp. 68720-68736, 2021, doi: 10.1109/ACCESS.2021.3077313.

[2] J. Song, F. Yang, W. Zhang and Z. Ma, "Parametric Model for Video Streaming Services With Different Spatial and Temporal Resolutions," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 9, pp. 3380-3390, Sept. 2021, doi: 10.1109/TCSVT.2020.3041489.

[3] S. Park, A. Bhattacharya, Z. Yang, S. R. Das and D. Samaras, "Mosaic: Advancing User Quality of Experience in 360-Degree Video Streaming With Machine Learning," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 1000-1015, March 2021, doi: 10.1109/TNSM.2021.3053183.

[4] N. Barman and M. G. Martini, "User Generated HDR Gaming Video Streaming: Dataset, Codec Comparison, and Challenges," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 3, pp. 1236-1249, March 2022, doi: 10.1109/TCSVT.2021.3077384.

[5] J. Hu, X. Liao, W. Wang and Z. Qin, "Detecting Compressed Deepfake Videos in Social Networks Using Frame-Temporality Two-Stream Convolutional Network," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 3, pp. 1089-1102, March 2022, doi: 10.1109/TCSVT.2021.3074259.

[6] A. Erfanian, F. Tashtarian, A. Zabrovskiy, C. Timmerer and H. Hellwagner, "OSCAR: On Optimizing Resource Utilization in Live Video Streaming," in IEEE Transactions on Network and Service Management, vol. 18, no. 1, pp. 552-569, March 2021, doi: 10.1109/TNSM.2021.3051950.

[7] C. G. Bampis, Z. Li, I. Katsavounidis, T. -Y. Huang, C. Ekanadham and A. C. Bovik, "Towards Perceptually Optimized Adaptive Video Streaming-A Realistic Quality of Experience Database," in IEEE Transactions on Image Processing, vol. 30, pp. 5182-5197, 2021, doi: 10.1109/TIP.2021.3073294.

[8] M. Hu, J. Chen, D. Wu, Y. Zhou, Y. Wang and H. -N. Dai, "TVG-Streaming: Learning User Behaviors for QoE-Optimized 360-Degree Video Streaming," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 10, pp. 4107-4120, Oct. 2021, doi: 10.1109/TCSVT.2020.3046242.

[9] W. Zou, W. Zhang and F. Yang, "Modeling the Perceptual Quality for Viewport-Adaptive Omnidirectional Video Streaming Considering Dynamic Quality Boundary Artifact," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 31, no. 11, pp. 4241-4254, Nov. 2021, doi: 10.1109/TCSVT.2021.3050157.

[10] D. Li, R. Du, A. Babu, C. D. Brumar and A. Varshney, "A Log-Rectilinear Transformation for Foveated 360-degree Video Streaming," in IEEE Transactions on Visualization and Computer Graphics, vol. 27, no. 5, pp. 2638-2647, May 2021, doi: 10.1109/TVCG.2021.3067762.

[11] Q. Jiang, V. C. M. Leung and H. Tang, "Statistical QoS-Guaranteed Traffic Rate Adaptation for Wireless Scalable Video Streaming," in IEEE Systems Journal, vol. 16, no. 2, pp. 3433-3436, June 2022, doi: 10.1109/JSYST.2022.3144218.

[12] S. Bayhan, S. Maghsudi and A. Zubow, "EdgeDASH: Exploiting Network-Assisted Adaptive Video Streaming for Edge Caching," in IEEE Transactions on Network and Service Management, vol. 18, no. 2, pp. 1732-1745, June 2021, doi: 10.1109/TNSM.2020.3037147.

[13] L. Sun, T. Zong, S. Wang, Y. Liu and Y. Wang, "Towards Optimal Low-Latency Live Video Streaming," in IEEE/ACM Transactions on Networking, vol. 29, no. 5, pp. 2327-2338, Oct. 2021, doi: 10.1109/TNET.2021.3087625.

[14] L. Cui, D. Su, S. Yang, Z. Wang and Z. Ming, "TCLiVi: Transmission Control in Live Video Streaming Based on Deep Reinforcement Learning," in IEEE Transactions on Multimedia, vol. 23, pp. 651-663, 2021, doi: 10.1109/TMM.2020.2985631.

[15] Y. Yu and S. Lee, "Remote Driving Control With Real-Time Video Streaming Over Wireless Networks: Design and Evaluation," in IEEE Access, vol. 10, pp. 64920-64932, 2022, doi: 10.1109/ACCESS.2022.3183758.

[16] M. Tang and V. W. S. Wong, "Online Bitrate Selection for Viewport Adaptive 360-Degree Video Streaming," in IEEE Transactions on Mobile Computing, vol. 21, no. 7, pp. 2506-2517, 1 July 2022, doi: 10.1109/TMC.2020.3038710.

[17] L. R. Jiménez, M. Solera, M. Toril, S. Luna-Ramírez and J. L. Bejarano-Luque, "The Upstream Matters: Impact of Uplink Performance on YouTube 360° Live Video Streaming in LTE," in IEEE Access, vol. 9, pp. 123245-123259, 2021, doi: 10.1109/ACCESS.2021.3110284.

[18] H. Xu, Z. Cai, D. Takabi and W. Li, "Audio-Visual Autoencoding for Privacy-Preserving Video Streaming," in IEEE Internet of Things Journal, vol. 9, no. 3, pp. 1749-1761, 1 Feb.1, 2022, doi: 10.1109/JIOT.2021.3089080.

[19] G. Zhang, J. Y. B. Lee, K. Liu, H. Hu and V. Aggarwal, "A Unified Framework for Flexible Playback Latency Control in Live Video Streaming," in IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 12, pp. 3024-3037, 1 Dec. 2021, doi: 10.1109/TPDS.2021.3083202.

[20] N. Kan, J. Zou, C. Li, W. Dai and H. Xiong, "RAPT360: Reinforcement Learning-Based Rate Adaptation for 360-Degree Video Streaming With Adaptive Prediction and Tiling," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 3, pp. 1607-1623, March 2022, doi: 10.1109/TCSVT.2021.3076585.

[21] C. Zheng, J. Yin, F. Wei, Y. Guan, Z. Guo and X. Zhang, "STC: FoV Tracking Enabled High-Quality 16K VR Video Streaming on Mobile Platforms," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 4, pp. 2396-2410, April 2022, doi: 10.1109/TCSVT.2021.3094909.

[22] Y. Hao, W. Wang and Q. Lin, "Incident Retrieval and Recognition in Video Stream Using Wi-Fi Signal," in IEEE Access, vol. 9, pp. 100208-100222, 2021, doi: 10.1109/ACCESS.2021.3096527.

[23] X. Zhang, X. Wei, L. Zhou and Y. Qian, "Social-Content-Aware Scalable Video Streaming in Internet of Video Things," in IEEE Internet of Things Journal, vol. 9, no. 1, pp. 830-843, 1 Jan.1, 2022, doi: 10.1109/JIOT.2021.3112200.

[24] R. Wang, L. Si and B. He, "Sliding-Window Forward Error Correction Based on Reference Order for Real-Time Video Streaming," in IEEE Access, vol. 10, pp. 34288-34295, 2022, doi: 10.1109/ACCESS.2022.3162217.

[25] X. Chen, T. Tan, G. Cao and T. F. L. Porta, "Context-Aware and Energy-Aware Video Streaming on Smartphones," in IEEE Transactions on Mobile Computing, vol. 21, no. 3, pp. 862-877, 1 March 2022, doi: 10.1109/TMC.2020.3019341.

[26] Q. Cheng, H. Shan, W. Zhuang, L. Yu, Z. Zhang and T. Q. S. Quek, "Design and Analysis of MEC- and Proactive Caching-Based $360^{\circ }$ Mobile VR Video Streaming," in IEEE Transactions on Multimedia, vol. 24, pp. 1529-1544, 2022, doi: 10.1109/TMM.2021.3067205.

[27] A. Zhang et al., "Video Super-Resolution and Caching—An Edge-Assisted Adaptive Video Streaming Solution," in IEEE Transactions on Broadcasting, vol. 67, no. 4, pp. 799-812, Dec. 2021, doi: 10.1109/TBC.2021.3071010.

[28] Z. Shang, J. P. Ebenezer, Y. Wu, H. Wei, S. Sethuraman and A. C. Bovik, "Study of the Subjective and Objective Quality of High Motion Live Streaming Videos," in IEEE Transactions on Image Processing, vol. 31, pp. 1027-1041, 2022, doi: 10.1109/TIP.2021.3136723.

[29] Q. Li et al., "A Super-Resolution Flexible Video Coding Solution for Improving Live Streaming Quality," in IEEE Transactions on Multimedia, vol. 25, pp. 6341-6355, 2023, doi: 10.1109/TMM.2022.3207580.

[30] D. Tanjung, J. -D. Kim, D. -H. Kim, J. Lee, S. Kim and J. -Y. Jung, "QoE Optimization in DASH-Based Multiview Video Streaming," in IEEE Access, vol. 11, pp. 83603-83614, 2023, doi: 10.1109/ACCESS.2023.3300380.

[31] J. Hu, X. Liao, W. Wang and Z. Qin, "Detecting Compressed Deepfake Videos in Social Networks Using Frame-Temporality Two-Stream Convolutional Network," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 3, pp. 1089-1102, March 2022, doi: 10.1109/TCSVT.2021.3074259.

[32] J. Chakareski, X. Corbillon, G. Simon and V. Swaminathan, "User Navigation Modeling, Rate-Distortion Analysis, and End-to-End Optimization for Viewport-Driven 360$^\circ $ Video Streaming," in IEEE Transactions on Multimedia, vol. 25, pp. 5941-5956, 2023, doi: 10.1109/TMM.2022.3201397.

[33] M. A. Khan et al., "A Survey on Mobile Edge Computing for Video Streaming: Opportunities and Challenges," in IEEE Access, vol. 10, pp. 120514-120550, 2022, doi: 10.1109/ACCESS.2022.3220694.

[34] Y. Li, X. Zhang, C. Cui, S. Wang and S. Ma, "Fleet: Improving Quality of Experience for Low-Latency Live Video Streaming," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 33, no. 9, pp. 5242-5256, Sept. 2023, doi: 10.1109/TCSVT.2023.3243901.

[35] Y. Tang, Y. Wu, P. Zhou and J. Hu, "Enabling Weakly Supervised Temporal Action Localization From On-Device Learning of the Video Stream," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 41, no. 11, pp. 3910-3921, Nov. 2022, doi: 10.1109/TCAD.2022.3197536.

[36] R. Farahani, M. Shojafar, C. Timmerer, F. Tashtarian, M. Ghanbari and H. Hellwagner, "ARARAT: A Collaborative Edge-Assisted Framework for HTTP Adaptive Video Streaming," in IEEE Transactions on Network and Service Management, vol. 20, no. 1, pp. 625-643, March 2023, doi: 10.1109/TNSM.2022.3210595.

[37] G. Zhang, K. Liu, H. Hu, V. Aggarwal and J. Y. B. Lee, "Post-Streaming Wastage Analysis – A Data Wastage Aware Framework in Mobile Video Streaming," in IEEE Transactions on Mobile Computing, vol. 22, no. 1, pp. 389-401, 1 Jan. 2023, doi: 10.1109/TMC.2021.3069764.

[38] X. Yuan, L. Pu, J. Shi, Q. Gong and J. Xu, "Muster: Multi-Source Streaming for Tile-Based 360° Videos Within Cloud Native 5G Networks," in IEEE Transactions on Mobile Computing, vol. 22, no. 11, pp. 6616-6632, 1 Nov. 2023, doi: 10.1109/TMC.2022.3194101.

[39] R. R. Ramachandra Rao, S. Göring and A. Raake, "AVQBits—Adaptive Video Quality Model Based on Bitstream Information for Various Video Applications," in IEEE Access, vol. 10, pp. 80321-80351, 2022, doi: 10.1109/ACCESS.2022.3195527.

[40] K. Park, M. Kim and L. Park, "NeuSaver: Neural Adaptive Power Consumption Optimization for Mobile Video Streaming," in IEEE Transactions on Mobile Computing, vol. 22, no. 11, pp. 6633-6646, 1 Nov. 2023, doi: 10.1109/TMC.2022.3195961.

[41] G. Zhang et al., "DUASVS: A Mobile Data Saving Strategy in Short-Form Video Streaming," in IEEE Transactions on Services Computing, vol. 16, no. 2, pp. 1066-1078, 1 March-April 2023, doi: 10.1109/TSC.2022.3150012.

[42] N. Barman and M. G. Martini, "User Generated HDR Gaming Video Streaming: Dataset, Codec Comparison, and Challenges," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 3, pp. 1236-1249, March 2022, doi: 10.1109/TCSVT.2021.3077384.

[43] C. Zheng, J. Yin, F. Wei, Y. Guan, Z. Guo and X. Zhang, "STC: FoV Tracking Enabled High-Quality 16K VR Video Streaming on Mobile Platforms," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 4, pp. 2396-2410, April 2022, doi: 10.1109/TCSVT.2021.3094909.

[44] C. Gao, Y. Wang, Y. Han, W. Chen and L. Zhang, "IVP: An Intelligent Video Processing Architecture for Video Streaming," in IEEE Transactions on Computers, vol. 72, no. 1, pp. 264-277, 1 Jan. 2023, doi: 10.1109/TC.2022.3155950.

[45] R. Avanzato, F. Beritelli and C. Rametta, "Enhancing Perceptual Experience of Video Quality in Drone Communications by Using VPN Bonding," in IEEE Embedded Systems Letters, vol. 15, no. 1, pp. 1-4, March 2023, doi: 10.1109/LES.2022.3182466.

[46] S. Lee, J. -B. Jeong and E. -S. Ryu, "Group-Based Adaptive Rendering System for 6DoF Immersive Video Streaming," in IEEE Access, vol. 10, pp. 102691-102700, 2022, doi: 10.1109/ACCESS.2022.3208599.

[47] D. Y. Lee et al., "A Subjective and Objective Study of Space-Time Subsampled Video Quality," in IEEE Transactions on Image Processing, vol. 31, pp. 934-948, 2022, doi: 10.1109/TIP.2021.3137658.

[48] N. Kan, J. Zou, C. Li, W. Dai and H. Xiong, "RAPT360: Reinforcement Learning-Based Rate Adaptation for 360-Degree Video Streaming With Adaptive Prediction and Tiling," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 32, no. 3, pp. 1607-1623, March 2022, doi: 10.1109/TCSVT.2021.3076585.

[49] C. He, R. G. d. A. Azevedo, J. Chen, S. Zhu, B. Zeng and P. Frossard, "Quality-Constrained Encoding Optimization for Omnidirectional Video Streaming," in IEEE Transactions on Circuits and Systems for Video Technology, vol. 33, no. 10, pp. 6185-6190, Oct. 2023, doi: 10.1109/TCSVT.2023.3245066.

[50] D. Y. Lee, J. Kim, H. Ko and A. C. Bovik, "Video Quality Model of Compression, Resolution and Frame Rate Adaptation Based on Space-Time Regularities," in IEEE Transactions on Image Processing, vol. 31, pp. 3644-3656, 2022, doi: 10.1109/TIP.2022.3173810.

[51] J. Chen, Z. Luo, Z. Wang, M. Hu and D. Wu, "Live360: Viewport-Aware Transmission Optimization in Live 360-Degree Video Streaming," in IEEE Transactions on Broadcasting, vol. 69, no. 1, pp. 85-96, March 2023, doi: 10.1109/TBC.2023.3234405.

[52] J. Tu, C. Chen, Z. Yang, M. Li, Q. Xu and X. Guan, "PSTile: Perception-Sensitivity-Based 360° Tiled Video Streaming for Industrial Surveillance," in IEEE Transactions on Industrial Informatics, vol. 19, no. 9, pp. 9777-9789, Sept. 2023, doi: 10.1109/TII.2022.3216812.