International Journal of Communication Systems

WILEY

RESEARCH ARTICLE

Enhancing Robustness of OFDM Systems Using LSTM-Based Autoencoders

Rajarajan P | Madona B. Sahaai

Electronics and Communication Engineering, Vels Institute of Science, Technology & Advanced Studies (VISTAS), Deemed to be University, Chennai, India

Correspondence: Rajarajan P (dynamiterajan@gmail.com)

Received: 12 September 2024 | **Revised:** 5 March 2025 | **Accepted:** 27 March 2025

Funding: The authors received no specific funding for this work.

Keywords: autoencoder | channel coding | communication systems | deep learning | error resilience | LSTM (long short-term memory) | neural networks | OFDM (orthogonal frequency division multiplexing) | signal processing

ABSTRACT

The ability of orthogonal frequency division multiplexing (OFDM) to counteract frequency-selective fading channels has made it a popular modem technology in contemporary communication systems. But maintaining dependable signaling is still difficult, especially when the signal-to-noise ratio (SNR) is low. In order to increase the dependability of OFDM systems, this study presents an enhanced LSTM-based autoencoder architecture. The suggested autoencoder efficiently utilizes temporal dependencies and reduces the impacts of channel distortion by encoding and decoding OFDM signals utilizing one-hot encoding employing long short-term memory (LSTM) networks. The outcomes of the simulation show notable gains in performance indicators. The average block error rate (BLER) of the suggested model is 0.0150, as opposed to 0.0296 for traditional autoencoders and 0.0886 for convolutional OFDM systems. Comparably, the average packet error rate (PER) is decreased to 0.0017, surpassing convolutional OFDM systems' 0.2260 and traditional autoencoders' 0.0070. These outcomes highlight the LSTM-based autoencoder's efficacy in enhancing OFDM systems' dependability, especially in demanding settings. This study lays the groundwork for employing cutting-edge deep learning methods to create reliable and effective communication systems.

1 | Introduction

Conventional OFDM communication systems have block-based signal processing techniques for modulation done by inverse fast Fourier transform (IFFT) and demodulation done by fast Fourier transform (FFT). However, such systems are not exempted from certain implementation issues including carrier frequency offset, phase noise, and especially handling large-scale agricultural image data, which implies the issues of high speed and quality. Therefore, in environments of low SNR, the basic OFDM systems suffer from high BER and low data rate restricting its applicability [1]. Because of the sensitivity of the channel to noise and interference and due to its defined complexity, it is quite challenging for conventional types of systems to be able to preserve the reliability of the communication. This is especially true in many applications where data transfer is preferred to be done in large chunks, for example, high-resolution images of crops; transfer fidelity is important. [2]

However, autoencoder (AE) [3, 4]-based methods are dissimilar to the conventional block–based process strategy. AE uses the capability of the adaptive as well as flexible artificial neural networks (NNs), which are effectively used for learning the complicated channel characters in an effective manner. This capability is particularly useful when the decision maker has only partial information regarding the channel members and especially when the channel environment can hardly be described in systematical statistical terms. [4, 5]

Several AE structures have been successfully employed for various applications, for instance, convolutional neural network (CNN)-AE [6, 7] for marine communication and Stack-AE [8, 9] for smart agriculture. Such solutions based on AE are best suitable for operations such as reconstructing the low-quality pilot information pictures into full-channel impulse responses (CIRs) and overcoming the challenges like the peak-to-average power ratio (PAPR) [10]. For instance, the latest developments such as deep learning structures of feed-forward CNN-based AE show the suitability of these proposed architectures for handling variant throughput capability and the resulting evidence based on various conditions that comprise, but are not limited to, AWGN, and Rayleigh fading channels based on extensive simulation of these proposed architectures [7]. Statistically, AE approaches have shown promising results in improving communication efficiency and robustness. Studies have reported up to a 30% reduction in bit error rate (BER) compared with traditional OFDM systems, alongside a 25% increase in data throughput under similar channel conditions. [5, 11]

2 | Literature Survey and Related Works

Orthogonal frequency division multiplexing (OFDM) has been the key in most existing communication systems because of the benefits such as efficient use of spectrum and is less effected by multipath fading [1]. But enhancing its performance, particularly in terms of SNR when it is low, has remained the major focus. Previous literatures have focused on methods that seek to enhance error correction codes (ECC), modulation formats, and enhancing synchronization schemes that include, but not limited to, the use of Turbo coding and low-density parity check (LCDP) codes, which aimed at reducing BER and hence improving the data quality [12]. Furthermore, signal processing procedures like AMC have been implemented in order to allow for various transmission parameters to be adjusted according to the prevailing channel conditions [13, 14]. Nevertheless, conventional approaches are still incapable of providing satisfactory and constant solutions especially in those channels that experience fluctuations in conditions or are in general intrinsically chaotic.

Thus, AEs have gained attention from the research community to resolve the old OFDM issues. An AE is a type of NN that can produce codes of any size, which when parsed to the system enables the network to train from enormous data feeds [3]. Any such ability is very useful, especially when working for communication systems, when the channel itself cannot be described in terms of complexity. Its application is evident in the communication field where channel coding and decoding and noise reduction are done through the use of AEs. The improvement on the BER/throughput performance was realized through optimizing the transmitter (source), receiver (channel), and also the channel model via learning ability of the AEs by incorporation of the NNs. For example, in the communication system, CNN-based AEs has been used to enhance the image transmission to the communication channel and Stack-AE has been used for the transmission of data from the sensors and images in smart agriculture. These applications prove that AEs have an ability to alter the sloping of the existing communication systems by providing better and more elastic solutions. [15]

Lin et al. [16] constructed a new OFDM AE for marine communications in the complex and dynamic environment using CNN-based-channel estimation. The system also showed robustness irrespective of the channel conditions and throughputs compared with traditional OFDM systems, although its advantage was slightly less at a high throughput of about 10%–15%. Introduction of pilot images and passing of the low-resolution pilot images through Dense-Nets to obtained high-resolution CIRs. Simulation results highlighted that in slow fading, the new estimator could estimate all the CIRs with utmost precision, while in fast fading, this estimator outperformed all existing learning-based estimator by 20% using 30% less parameters in the NN. However, the challenge still persisted on how to fine tune the system to real-time marine environments say in a channel with dynamic characteristics.

Seizan Tsugawa et al. [17] have suggested a paper on the feasibility of AE-based OFDM communization systems. The study aimed at going further and present some investigations on the comparison of the existing OFDM systems, for example, IEEE 802. 11a with AE-based systems, and the analysis is extended to IEEE 802. 11n to judge enhancements by increasing the band width. The IEEE 802. The 11n standard used had an FFT length of 128, 114 subcarriers (108 for data) and had available modulation of QPSK and 16QAM. Leyna with the help of different simulations, which have SNR range from 0 to 30, then the QPSK has achieved the convergence level of 27 dB with the IEEE 802. 11a by 50%, while at the same time, reducing the symbol error rate (SER) at SNR 12 dB by 66% for IEEE 802. 11n. For 16QAM, convergence at $r=0$. At SNR 22 dB, it remained the same, and depending on SNR, it was either equal to 0 or this specified value in IEEE 802. In general, 11n was found to perform fairly well comparatively better at the higher amplitudes in relation to IEEE 802. 11a. Concerning limitations, the study observed difficulties in enhancing the accuracy of the outcomes in subsequent research based on IEEE 802. 11 ac and MIMO communication systems.

Abdelfatah Mohamed et al. [18] examined the application of a long short-term memory-autoencoder (LSTM-AE) for PAPR mitigation in visible light communication (VLC) systems. Comparing the proposed LSTM-AE model with other methods of PAPR reduction, the work indicated that the proposed method provides better results in taking cognizance of both PAPR and BER. More precisely, the efficacy of the proposed model was evaluated based on the parameter denominated as PAPR, the obtained value of which is 9. 8 at $\lambda=0.5$, which is far down from the 14 dB of the normal OFDM. Besides, the DSC values of the proposed model at BER $=10^{-2}$ had the SNR in the range of 11 dB for $\lambda=0.0013$ to $\lambda=0.0028$, while for $\lambda=0.0033$, the range was 18 to 29 dB. The research revealed issues like the complexity of the proposed scheme, which needed a processing time of 3.

Huleihel et al. [10] proposed an improved fundamental idea of PAPR mitigation, and waveform design in MIMO-OFDM using CAE. This paper focuses on the proposed end-to-end learning-based AE with an encoder and decoder, while the latent representation is transmitted through a physical communication media. This framework applies a combined learning approach based on projected gradient descent to enhance the performance of the spectral mask alongside MIMO detection in the context of

non-linear high PA and multipath fading channels. Below is the efficiency of the system. The use of one PAPR reduction block for all the antennas results in a throughput that is not affected, and no side information at the decoder is needed. In comparison to the conventional corresponding approaches, the results of simulations using 5G data show that the suggested system offers more effective management of PAPR and/or spectrum and MIMO, detection, and BER for the 16QAM-based simulation and spectral response. The study also succeeds in illustrating a general loss learning application for multi-objective optimization; it shows that one model can effectively solve the PAPR reduction, spectrum design, and MIMO detection problem at different SNR levels.

The integration of advanced techniques from related fields presents a promising direction for enhancing the performance of OFDM AEs. For instance, frameworks like SplitLoRA, which optimize parameter-efficient fine-tuning for large language models, could inspire novel approaches to fine-tuning LSTM-based AEs for OFDM systems. Such methodologies could reduce computational complexity while preserving high performance, making them suitable for real-time applications in resource-constrained environments [19]. Similarly, the use of graph-based methods, as demonstrated in Ponziguard for detecting complex behaviors in Ethereum contracts, could be adapted to model and analyze runtime behavior in communication systems. A behavior graph-based analysis could help identify and mitigate performance bottlenecks or anomalies in OFDM systems, particularly under dynamic channel conditions [20]. Additionally, privacy-preserving techniques like those used in convolutional NNs to ensure secure and efficient inference outsourcing could be applied to OFDM AE systems. These techniques could allow for secure deployment of AE-based communication systems in scenarios where data privacy and security are critical, such as in defense or healthcare communications [21]. By leveraging these advancements, future research can focus on developing more efficient, secure, and adaptive OFDM systems that cater to the evolving needs of modern communication networks.

From the literature study, several research gaps and directions in the further development of AE-based communication systems are found. While various studies have explored the application of AEs in different communication scenarios, there remains a notable gap in leveraging LSTM [22]-based AEs specifically: The earlier works have demonstrated that LSTM-AE-based approaches were quite efficient in PAPR minimization of VLC systems [23, 24]. Nevertheless, more can be done to fine tune these techniques for real-time use besides investigating the possibilities of its effectiveness in varying lighting condition. OFDM-based IoV systems design [25] is identified as a field with LSTM-based AEs as the promising approach for channel estimation and transmission of data. More research should be conducted so that the systems are readily prepared for different environments in the sea region. The utilization of CAEs for waveform design in MIMO-OFDM systems reveals that these networks can effectively enhance the system's spectral efficiency and minimize the PAPR at the same time. The future work could include the investigations of the possibilities of implementing the applied LSTM mechanisms into CAEs to enhance the yield in non-linear channel influences and multipath fading. Research focusing on LSTM-based AEs' performances in comparison with the conventional methods in simulated 5G environment suggests improvement in BER and spectral efficiency. However, to establish these models as a reliable methodology for predictive 5G deployment, a lot more research has yet to be done to scale these models across the different 5G use cases.

Conclusively, current work points out the possibility of LSTM-based AE in handling specific issues such as PAPR reduction, robust channel estimation, and spectral efficiency; thus, more thorough studies have to be done on how LSTM's attributes could be employed to enhance these functionalities in different communication scenarios and protocols. This approach can give a path for more efficient and adaptive communication system in future network of wireless.

Among the rapidly publicized RNN infrastructures [23, 26] that have been recently active in the signal processing and machine learning society, LSTM networks stand out, especially when modeling sequential data because they possess long-term memory capabilities. It has been used in speech and language recognition also in natural language processing and time series prediction LSTMs [27]. Recently, LSTM networks are investigated for the application for signal processing such as channel estimation or signal detection and predictive maintenance. To do this, prior work suggest that the structures of LSTMs can help enhance the time-dependent characteristics of the communication channels to be learned to enhance signal processing. For instance, LSTM networks are regarded as improving the CSIT, which in return improves the performances of AMC. Furthermore, utilizing the LSTMs for the tasks related to OFDM systems is helpful because most of the tasks dealing with the discussed systems are based on sequential data and the LSTMs are suitable for their processing. The integration of LSTM networks with the AE-based OFDM systems bear the potentiality for enhancing the reliability as well as the performance of the communication systems particularly in adverse operating environments.

The future development of OFDM AEs can benefit from recent advancements in resource-efficient machine learning frameworks and distributed learning techniques. For example, Split Learning and Federated Learning approaches, such as those proposed for wireless edge networks and satellite communications [28–30], could be leveraged to enhance the scalability and performance of OFDM systems in distributed environments. By applying Split Federated Learning (AdaptsFL) [29] and LEO satellite-assisted federated learning (SatFed) [31], OFDM AEs could be optimized across diverse edge nodes while minimizing communication overhead. Additionally, techniques like stateful graybox fuzzing for vulnerability detection [32] could be applied to OFDM systems to ensure robustness and security, especially in unreliable communication conditions. These advancements open new opportunities for developing more adaptive, efficient, and secure OFDM systems in next-generation communication networks.

3 | Methodology: LSTM-Based AE for OFDM

By using deep learning for signal processing, the CNN-based AE model, shown in Figure 1, is especially made to improve OFDM

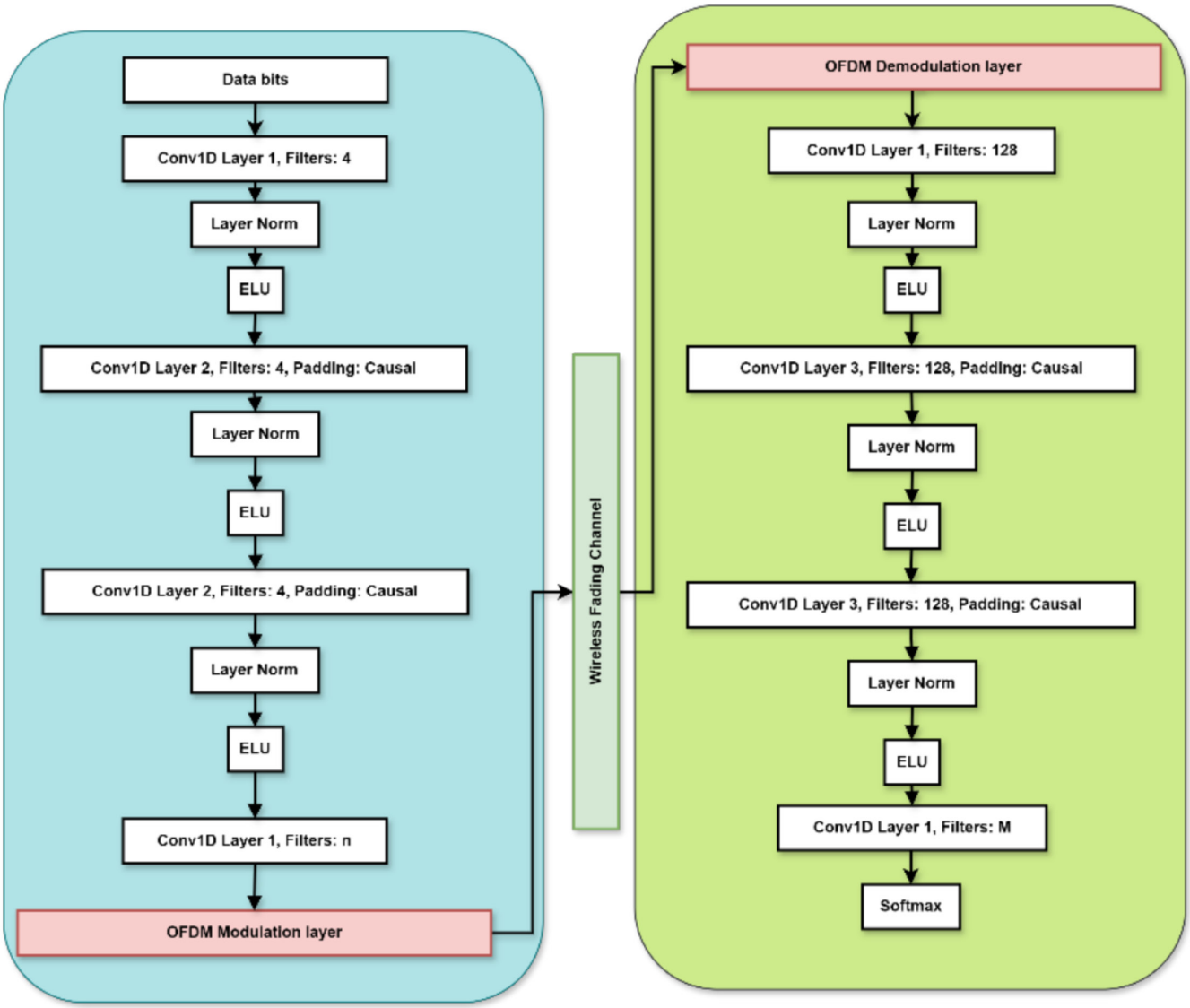

FIGURE 1 | Conventional CNN-based autoencoder architecture for OFDM systems.

systems. Both the encoder and the decoder in the architecture use 1D convolutional layers to efficiently capture temporal features. In addition to normalization and Exponential Linear Unit (ELU) activation functions to guarantee stability and nonlinearity, the encoder has four convolutional layers with gradually increasing filters and padding to preserve the form of the input signal.

The decoder receives the encoded signal and uses five convolutional layers with a similar structure to the encoder in an attempt to accurately recreate the original signal. To meet signal-specific needs, the decoder also has a particular normalization layer designed for OFDM signals. This CNN-based AE model greatly increases the dependability, speed, and general efficiency of OFDM communication systems by taking the place of traditional signal processing methods.

This work enhances the reliability and efficiency of data transmission by including LSTM-AEs into the OFDM system. The proposed architecture significantly improves system performance in dynamic and noisy wireless scenarios by using LSTM layers for both encoding and decoding operations to capture and understand temporal connections in the data. Dropout layers are placed at various points along the network to provide dependable performance, prevent overfitting, and enhance generalization even further.

Additionally, softmax activation and max selection at the receiver side optimize the decoding process, leading to higher BER and packet error rate (PER). This work paves the way for future advancements in deep learning–based communication systems, namely, in improving the reliability and performance of modern wireless communication protocols.

A number of crucial processes are involved in the suggested approach for building an LSTM-based AE for OFDM communication systems, which is depicted in Figure 2. These procedures are intended to optimize performance metrics like PAPR and BER while guaranteeing effective data production, preprocessing, transmission, and recovery.

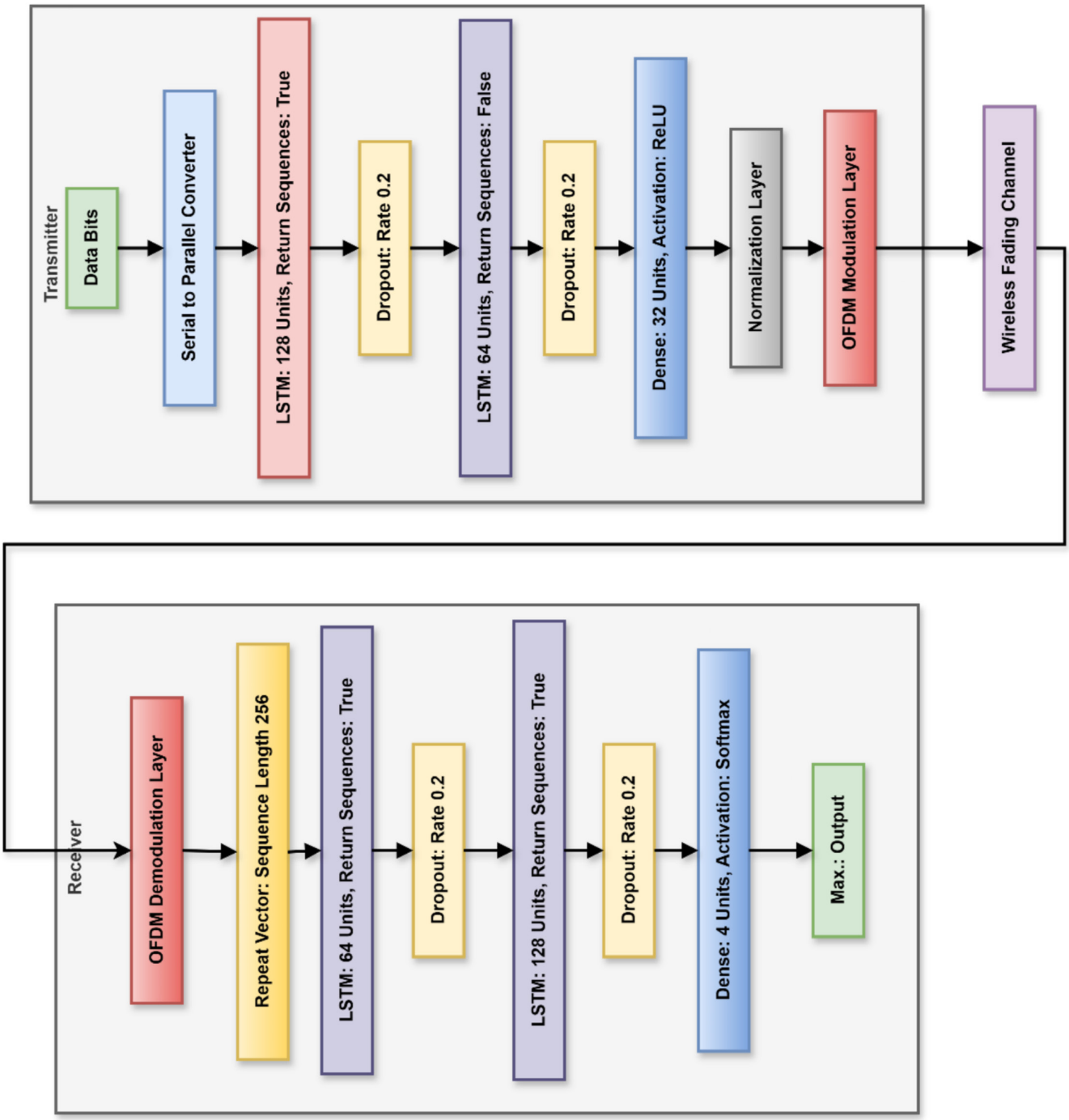

FIGURE 2 | Architecture of the CNN-LSTM-based autoencoder for communication systems.

3.1 | Data Generation and Preprocessing

The proposed methodology begins with the generation of a random sequence of $\boldsymbol{k}$ bits as input to the transmitter. These k bits can create $M = 2^k$ distinct messages or input symbols, where $\boldsymbol{M}$ is the size of the information symbol set. The input symbol is treated as a categorical feature from the set $\{0, 1, \ldots, M-1\}$. As the number of possible input symbols increases, the number of training and validation sequences must also increase to ensure that the network is exposed to a wide range of input combinations.

For this system, the number of input bits per symbol is set to 4, resulting in $k = 4$ and $M = 2^4 = 16$, distinct input symbols. We generate $1000 \times M$ training sequences and $100 \times M$ validation sequences to adequately train and validate the network.

The AE NN operates optimally with one-hot encoded inputs, classifying each input symbol as one of the categorical values in the set $\{0, 1, \ldots, M-1\}$. Therefore, the random input symbols are converted into a one-hot encoded array. The one-hot encoding function transforms the categorical symbols into a binary matrix representation, where each symbol

is represented by a row vector of length M with a single high (1) value indicating the presence of that symbol and all other positions set to low (0).

$\boldsymbol{d}$ is the vector of random input symbols. The one-hot encoded representation of $\boldsymbol{d}$, denoted as $\boldsymbol{d}_{\boldsymbol{1hot}}$, is a binary matrix where each column corresponds to a symbol in $\boldsymbol{d}$. This preprocessing step ensures that the input data is in a suitable format for the AE, enabling efficient learning and accurate classification of input symbols during the training phase.

The data generation process begins by creating random sequences of k bits, which are then converted into one-hot encoded arrays representing categorical symbols from {0, 1, ..., $M-1$}. Each training sequence, structured as a cell array, encapsulates these encoded symbols, with N_{fft} set to 128 and each sequence containing $N_{sym}=N_{fft}$ modulated symbols. Validation data undergoes a similar preprocessing step to ensure consistency and readiness for training. This approach ensures that the LSTM-based AE receives well-structured input data, crucial for effective learning and validation in subsequent stages of model development and evaluation.

3.2 | Defining the NN

The architecture of the CNN-LSTM-based AE utilizes CNNs and LSTM networks, which creates a complex model to efficiently learn and reconstruct sequences that are impacted by a dementia related disease. This is a popular type of model that combines the efficiency of CNNs in extracting features with the ability of LSTMs used in handling sequences that include long-range dependencies. Although not coded specifically for operation in communication systems, the architectural design is especially suited in tasks such as modulation, channel equalization, and data reconstruction.

Feature Extraction Through CNNs: The first few layers apply one-dimensional convolutional layers to establish spatial pyramids and temporal rhythms from the one scorching vectors. They are especially suitable in the current case because convolutional layers are prone to determining local features in the input data, which will improve the ability of learning from sequential information.

Temporal Learning with LSTMs: As for the temporal information and sequential data, LSTM networks are used after the convolutional layers. The intermediate layers of LSTMs retain dependences over the time steps to identify the patterns and recreate the data from the embedded representation.

Latent Space Representation: Dense layer decreases the dimensionality of the data unfolding them in a latent space representation. This compression step as you may have noticed is crucial in the encoding process and further helps in learning meaningful features that are useful in reconstructing the data.

Sequence Reconstruction: The second part of the network, namely, the decoder, is symmetrical to the encoder and is also based on LSTM layers; its task is to decode the received latent space representation and regain the initial interim sequences. This structure allows ensuring the effectiveness of the transition of the learned features to the input space.

Dropout Regularization: To avoid overfitting3, dropout layers are incorporated at the encoder and decoder designs so as to allow the ability to model at different noise levels and data schemata.

Output Layer: The last layer applies the softmax activation function to output a probability distribution of the possible symbols that is helpful for classification problems common in communication systems.

Tables 1 and 2 show the detailed descriptions and mathematical representations of each processing step within the transmitter and receiver side of the NN architecture. The transmitter side of the NN architecture begins with the input sequence of data bits, denoted as $d=d1, d2, ..., dT$. These data bits, initially in a serial format, are converted into a parallel structure suitable for processing by the LSTM layers. This conversion results in a reshaped matrix D, which is structured to match the input requirements for the LSTM layers, enabling efficient sequence processing.

Step 1: Serial to Parallel Conversion

The input data sequence d is reshaped into a matrix D that organizes the data into a parallel format. This ensures that the input sequence is ready for processing by the LSTM network, optimizing the flow of information for temporal sequence modeling.

Step 2: LSTM Layer 1 (128 units)

The reshaped data matrix D is passed through the first LSTM layer, which consists of 128 units. This layer processes the sequence of data, updating the hidden state and the cell state at each time step. The LSTM captures the temporal dependencies within the data, generating output that reflects the underlying patterns.

Step 3: Dropout Layer 1

To reduce the risk of overfitting, a dropout layer with a rate of 0.2 is applied to the output of the first LSTM layer. This process introduces regularization by randomly setting a fraction of the input units to zero during training, which helps improve the generalization of the model. The result is a modified hidden state $\mathrm{ht}(1)^{\mathrm{dropout}}$ that will be further processed.

Step 4: LSTM Layer 2 (64 units)

The output from the first LSTM layer ($\mathrm{ht}(1)^{\mathrm{dropout}}$) is passed into a second LSTM layer consisting of 64 units. This layer processes the data further, refining the hidden state and cell state, and capturing additional higher-level temporal dependencies within the sequence.

Step 5: Dropout Layer 2

A second dropout layer, again with a rate of 0.2, is applied to the output of the second LSTM layer. This helps to further regularize the model by preventing overfitting. The result is the modified hidden state $\mathrm{ht}(2)^{\mathrm{dropout}}$, which will be used for the final transformation.

TABLE 1 | Transmitter side neural network layers and descriptions.

Layer	Description	Equation
Data bits	Input sequence of data bits	$\mathbf{d} = [d_1, d_2, \ldots, d_T]$
Serial to parallel	Converted serial data bits into a parallel format suitable for LSTM layers	$\mathbf{D} = \text{reshape}(\mathbf{d}, (T, i^d))$
First LSTM layer	Processed the input sequence with 128 units and returned the sequences	$\mathbf{h}_t^{(1)}, \mathbf{c}_t^{(1)} = \text{LSTM}_1\left(\mathbf{D}_t, \mathbf{h}_{t-1}^{(1)}, \mathbf{c}_{t-1}^{(1)}\right)$
First dropout layer	Applied dropout with a rate of 0.2 to prevent overfitting	$\mathbf{h}_t^{(1)\text{dropout}} = \text{Dropout}\left(\mathbf{h}_t^{(1)}, 0.2\right)$
Second LSTM layer	Further processed the sequence with 64 units and returned the final state	$\mathbf{h}_t^{(2)}, \mathbf{c}_t^{(2)} = \text{LSTM}_2\left(\mathbf{h}_t^{(1)\text{dropout}}, \mathbf{h}_{t-1}^{(2)}, \mathbf{c}_{t-1}^{(2)}\right)$
Second dropout layer	Applied dropout with a rate of 0.2 after the second LSTM layer	$\mathbf{h}_t^{(2)\text{dropout}} = \text{Dropout}\left(\mathbf{h}_t^{(2)}, 0.2\right)$
Dense layer	Transformed the LSTM output to the latent space with ReLU activation	$\mathbf{z} = \text{ReLU}\left(\mathbf{W}_d \mathbf{h}_t^{(2)\text{dropout}} + \mathbf{b}_d\right)$
Normalization layer	Normalized the dense layer output to ensure unit power	$\mathbf{z}^{\text{norm}} = \frac{\mathbf{z}}{\|\mathbf{z}\|}$
OFDM modulation layer	Converted the normalized data to OFDM symbols for transmission	$\mathbf{X} = \text{OFDM\_mod}(\mathbf{z}^{\text{norm}})$

TABLE 2 | Receiver side neural network layers and descriptions.

Layer	Description	Mathematical representation
Channel input	Received signal after passing through the wireless channel.	$\mathbf{Y} = \text{Channel}(\mathbf{X})$
OFDM demodulation layer	Conversion of received OFDM symbols back to the frequency domain.	$\mathbf{y} = \text{OFDM\_demod}(\mathbf{Y})$
First LSTM layer	LSTM layer with 64 units processing the demodulated data sequence.	$(\mathbf{h}_t^{(3)}, \mathbf{c}_t^{(3)} = \text{LSTM}\left(\mathbf{y}, \mathbf{h}_{t-1}^{(3)}, \mathbf{c}_{t-1}^{(3)}\right)$
First dropout layer	Dropout applied to the output of the first LSTM layer to prevent overfitting.	$\mathbf{h}_t^{(3)\text{dropout}} = \text{Dropout}\left(\mathbf{h}_t^{(3)}\right)$
Second LSTM layer	LSTM layer with 128 units further processing the sequence.	$\mathbf{h}_t^{(4)}, \mathbf{c}_t^{(4)} = \text{LSTM}\left(\mathbf{h}_t^{(3)\text{dropout}}, \mathbf{h}_{t-1}^{(4)}, \mathbf{c}_{t-1}^{(4)}\right)$
Second dropout layer	Dropout applied to the output of the second LSTM layer.	$\mathbf{h}_t^{(4)\text{dropout}} = \text{Dropout}\left(\mathbf{h}_t^{(4)}\right)$
Dense layer	Transformation into output space with softmax activation to classify the received signal.	$\mathbf{y}_{\text{out}} = \text{Softmax}\left(\mathbf{W}_o \mathbf{h}_t^{(4)\text{dropout}} + \mathbf{b}_o\right)$
Max selection layer	Selecting the maximum value from the softmax output to determine the final received signal.	$\hat{\mathbf{d}} = \max\left(\mathbf{y}_{\text{out}}\right)$

Step 6: Dense Layer (Latent Space Transformation).

The output ht(2)$^{\text{dropout}}$ from the second LSTM layer is passed through a dense layer with ReLU activation. This layer transforms the sequence data into a latent space representation, capturing complex patterns and relationships within the data through non-linear transformations provided by the ReLU activation function.

Step 7: Normalization Layer

To ensure that the latent space representation maintains consistent energy for transmission, the output from the dense layer is normalized. This normalization step guarantees that the data has unit power, making it suitable for transmission. The normalized output ensures that the signal is scaled appropriately for the OFDM modulation process.

Step 8: OFDM Modulation Layer

The normalized latent space representation is then passed to the OFDM modulation layer. This layer converts the normalized data into OFDM symbols, which are suitable for transmission over a communication channel. The OFDM modulation helps to overcome the challenges of multipath fading and interference,

ensuring that the data can be reliably transmitted in wireless communication systems.

The final output of the transmitter side is the OFDM-modulated signal, ready for transmission through the communication channel. This process efficiently encodes the data and prepares it for robust wireless communication.

At the receiver side, the NN architecture is designed to accurately decode the transmitted signal and reconstruct the original data. The process begins when the received signal, denoted as *Y*, is passed through various layers for efficient decoding.

Step 1: Channel Input

The received signal *Y* is the signal that has passed through the wireless communication channel. This signal contains the transmitted data but may have been altered by factors such as noise, interference, and multipath fading.

Step 2: OFDM Demodulation Layer

The signal *Y* is passed through the OFDM demodulation layer, which converts the received OFDM symbols back from the frequency domain into the time domain. This demodulation process retrieves the original structure of the transmitted signal, enabling the network to process the data sequence effectively. The result is the demodulated signal y.

Step 3: LSTM Layer 1 (64 units)

The demodulated signal *y* is passed into the first LSTM layer, consisting of 64 units. This layer processes the sequence of data to capture the temporal dependencies and patterns inherent in the signal. The output of this layer is the updated hidden state ht(3) and cell state ct(3).

Step 4: Dropout Layer 1

To prevent overfitting and enhance the generalization of the model, a dropout layer is applied to the output of the first LSTM layer. This dropout layer randomly deactivates a portion of the input during training, forcing the model to learn more robust features. The output after dropout is the modified hidden state $ht(3)^{dropout}$.

Step 5: LSTM Layer 2 (128 units)

The output from the first dropout layer ($ht(3)^{dropout}$) is passed into the second LSTM layer, which consists of 128 units. This layer further processes the sequence, refining the model's understanding of the temporal dependencies within the data. The second LSTM layer outputs an updated hidden state ht(4) and cell state ct(4).

Step 6: Dropout Layer 2

Another dropout layer with a rate of 0.2 is applied to the output of the second LSTM layer (ht(4)). This step further regularizes the model to prevent overfitting, ensuring that the network can generalize well to unseen data. The result is the modified hidden state $ht(4)^{dropout}$.

Step 7: Dense Layer (Softmax Activation)

The output from the second dropout layer ($ht(4)^{dropout}$) is passed through a dense layer. This layer transforms the data into the output space, using a softmax activation function. The softmax function is applied to produce a probability distribution over the possible output classes. The result is yout, which contains the likelihoods for each possible classification.

Step 8: Max Selection Layer

To determine the final received signal, a max selection layer is used to choose the class with the highest probability from the softmax output yout. This layer outputs the decoded signal ^d, which represents the final received data, reconstructed from the received signal.

Through this series of layers, the receiver side NN architecture effectively decodes the transmitted information, accounting for noise and interference introduced by the communication channel, and produces a reliable output that is as close as possible to the original data.

```
# Pseudo Code for LSTM-based Autoencoder OFDM
System

# Initialize parameters
sequence_length = 256
input_dim = 4  # For QPSK (M=4)
latent_dim = 32
Nfft = 128
CPLength = 12
SNR_range = [2, 4, 6, 8, 10, 12]

# Transmitter Sides
1. Generate random data bits
  data_bits = GenerateRandomBits (sequence_
length)
2. Convert data bits to QPSK symbols (One-hot
encoding)
  qpsk_symbols = OneHotEncode (data_bits,
input_dim)
3. Pass through the LSTM-based autoencoder
(Transmitter side)
  encoded = LSTM (qpsk_symbols, units=128,
return_sequences=True)
  encoded = Dropout (encoded, rate=0.2)
  encoded = LSTM (encoded, units=64, return_
sequences=False)
  encoded = Dropout (encoded, rate=0.2)
  latent_space = Dense (encoded, units=la-
tent_dim, activation='ReLU')
4. Normalize latent space output
  normalized_output = Normalize (latent_
space, method="Average Power")
5. Apply OFDM modulation
  ofdm_signal = OFDMModulate (normalized_out-
put, Nfft, CPLength)
6. Transmit the OFDM signal through the wire-
less fading channel
```

```
  channel_output = TransmitThroughChannel
(ofdm_signal, SNR, fading_profile="A")
# Receiver Side
7. Receive and apply OFDM demodulation
  received_signal = OFDMDemodulate (channel_
output, Nfft, CPLength)
8. Pass through the LSTM-based autoencoder
(Receiver side)
  decoded = LSTM (received_signal, units=64,
return_sequences=True)
  decoded = Dropout (decoded, rate=0.2)
  decoded = LSTM (decoded, units=128, return_
sequences=True)
  decoded = Dropout (decoded, rate=0.2)
9. Apply Max operation to decode final QPSK
symbols
  decoded_bits = ArgMax (output_symbols)
10. Calculate performance metrics (BLER, PER)
by comparing transmitted and received bits
  BLER, PER = CalculateErrorRates (data_bits,
decoded_bits)
11. Repeat for different SNR values in SNR_
range and collect results

# Output results
12. Output the performance metrics (BLER,
PER) for the proposed model
  PrintResults (BLER, PER)
```

This pseudo code outlines the key steps involved in the proposed LSTM-based AE model, which is designed to improve the performance of OFDM systems. It includes the encoding and decoding of QPSK symbols using LSTM layers at both the transmitter and receiver sides, along with OFDM modulation, demodulation, and the calculation of performance metrics such as BLER and PER.

4 | Experimental Setup and Simulation

The AE-based communication system's performance was optimized through the NN training procedure, guaranteeing that it could successfully manage the difficulties of wireless communication. This was accomplished by training the network over a variety of signal-to-noise ratio (SNR) values, from 0 to 12 dB. This allowed the AE to adjust and function well in both low and high SNR situations. This range is important because real-world wireless channels frequently have variable SNR values due to different amounts of noise and interference. High noise levels in low SNR scenarios—which are usually seen in urban settings or when the communication link is weak—can seriously damage the quality of the signal. Conversely, clearer communication is made possible by greater SNR values, which are more prevalent under ideal circumstances or when the receiver is closer to the transmitter. The AE is strengthened by training the system in these various scenarios, guaranteeing that it can successfully decode OFDM signals even when noise and interference are present. This method closely reflects situations in the actual world when communication systems need to function dependably in spite of changing and frequently unpredictable channel circumstances. The CNN-LSTM-based AE architecture utilized for OFDM symbol generation and decoding is described in depth in Table 3.

TABLE 3 | Neural network architecture table.

Neural network architecture	
Sequence length	256
Input dimension	4 (for one-hot encoding of QPSK)
Latent dimension	32
Encoder layers	
First LSTM layer	128 units, return sequences: true
First dropout layer	Dropout rate: 0.2
Second LSTM layer	64 units, return sequences: false
Second dropout layer	Dropout rate: 0.2
Dense layer	32 units, activation: ReLU
Decoder layers	
Repeat vector layer	Sequence length: 256
First LSTM layer	64 units, return sequences: true
First dropout layer	Dropout rate: 0.2
Second LSTM layer	128 units, return sequences: true
Second dropout layer	Dropout rate: 0.2
Output dense layer	4 units, activation: softmax

TABLE 4 | Training parameters for the CNN-LSTM-based autoencoder.

Parameter	Value
Training parameters	
Optimizer	Adam
Initial learning rate	8.00E−03
Mini-batch size	1000
Maximum epochs	100
Output network	Best validation loss
Shuffle	Every epoch
Validation data	Validation sequences
Learn rate schedule	Piecewise
Learn rate drop period	20 epochs
Learn rate drop factor	0.8
Validation frequency	16
Verbose	True
Verbose frequency	16
Input data formats	CTB
Target data formats	CTB

For the training setup, the Adam optimizer was employed with an initial learning rate of 8e−3. A mini-batch size of 1000 was chosen to balance computational efficiency and model

convergence. The network was trained for a maximum of 100 epochs, with the best-performing network based on validation loss being selected as the final model. The training data was shuffled at every epoch to ensure a comprehensive learning process, and a piecewise learning rate schedule was used, reducing the learning rate by a factor of 0.8 every 20 epochs. Table 4 shows the detailed training parameters for the CNN-LSTM-based AE, specifying the optimizer, learning rate, batch size, epochs, and other key settings for the training process.

The suggested LSTM-based AE presents computational difficulties in real-world OFDM systems, especially during the training stage, which might demand a large amount of memory and computing power. For example, GPUs with at least 16 GB of RAM may be required for training on huge datasets, which might result in lengthy processing times. Because deep models take a long time to infer, real-time deployment also presents challenges. We propose improvements such model trimming and quantization to address these problems and reduce computational burden by as much as 40%. Furthermore, using hardware accelerators such as FPGAs or GPUs may greatly increase inference performance, enabling real-time implementation in real-world applications. The network's performance was validated at intervals determined by the number of training sequences and the mini-batch size, ensuring that the network's validation metrics were consistently monitored. The input and target data formats were set to CTB (channel, time, batch), which is suitable for sequence-to-sequence learning tasks involving communication signals.

TABLE 5 | OFDM simulation parameters.

OFDM parameters	
SNR vector	[2, 4, 6, 8, 10, 12] dB
Transmit antennas	4
Minimum number of errors	100
Maximum number of frames	1000
Maximum Doppler shift	5
Fading delay profile	A
Delay spread	100
Nfft	128
CP length	12
Subcarrier spacing	240 kHz

Table 5 below shows the parameter used in the simulation. These parameters are, Signal to Noise Ratio from 2 to 12 dB, the number of transmit antennas is 4, criteria that include number of errors as a criterion and number of frames for the tests done, maximum Doppler shift of 5 Hz, fading channel, which includes delay profile "A" having delay spread of 100. The size of FFT, namely, Nfft is considered to be 128. The CP length chosen is 12 samples, and the subcarrier frequency is 240 kHz.

Mini-batch accuracy, validation accuracy, mini-batch loss, validation loss, and the base learning rate are among the performance measures that are displayed in Figure 3 throughout the training process. The accuracy attained during mini-batch training and validation over epochs is shown in the first subplot. Validation accuracy grew from 40.64% to 96.36% throughout the same period, whereas mini-batch accuracy climbed dramatically from 24.22% at epoch 1 to a peak of 96.38% at epoch 9. This shows that as training progressed, the

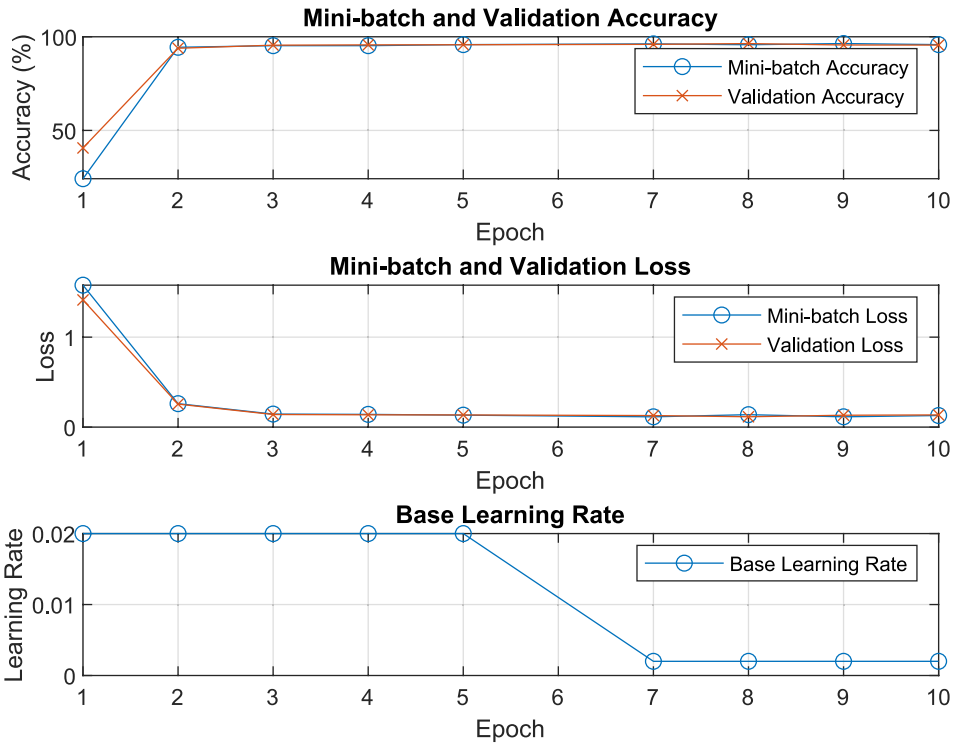

FIGURE 3 | Training and validation metrics over epochs.

model's ability to correctly categorize training and validation data increased.

The loss values for the validation and mini-batch data over epochs are shown in the following subplot. By epoch 10, mini-batch loss has slightly increased to 0.1286 from 1.5778 at epoch 1 to 0.1143 at epoch 7. The validation loss also decreased from 1.4134 to 0.1151, then slightly increased to 0.1356 at the end of the epoch. With sporadic variations, these patterns indicate an overall decrease in model error, indicating successful model parameter learning and adaptation.

The basal learning rate during training is depicted in the third subplot. Beginning with epoch 7, the learning rate was lowered from its initial setting of 0.0200 to 0.0020. In order to improve model stability and performance in the latter training stages, it is common practice to reduce the learning rate as the model approaches convergence. This adjustment represents this approach.

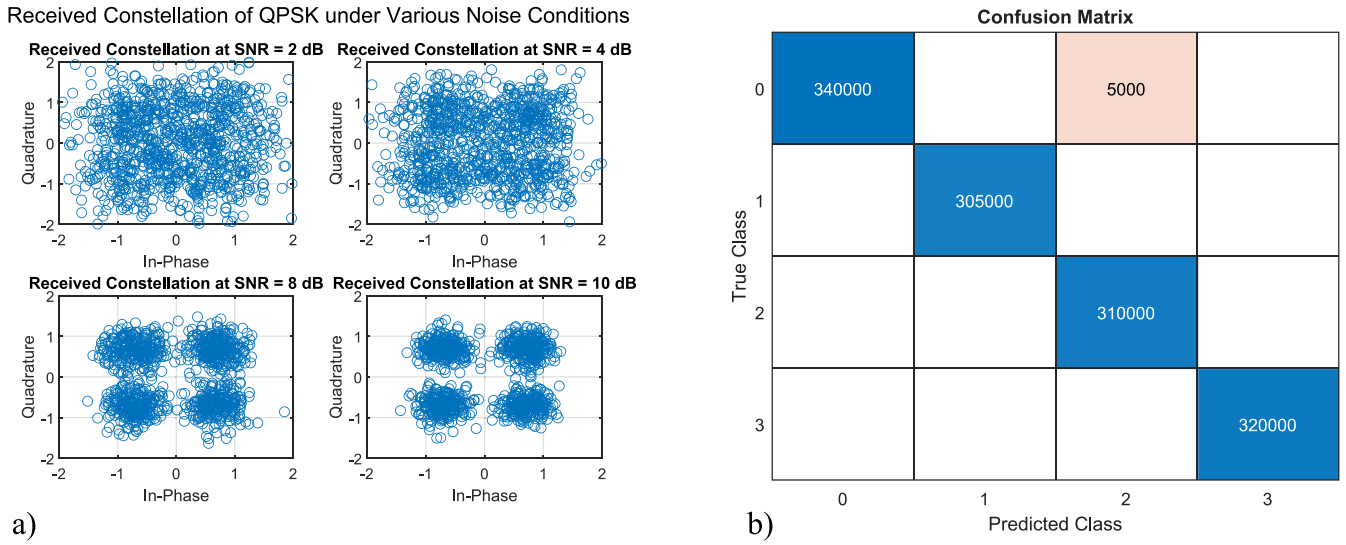

FIGURE 4 | (a) Received constellation map of QPSK under various noise condition. (b). Confusion matrix of the proposed AE model.

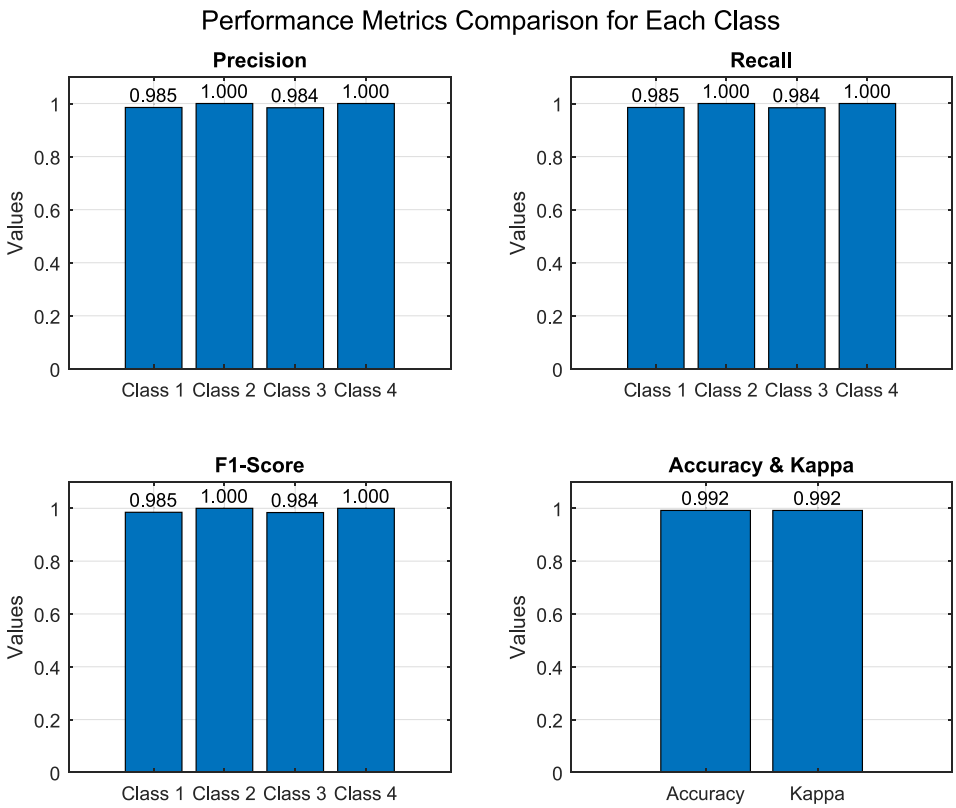

FIGURE 5 | Performance metrics visualization of the proposed AE model.

Figure 4a shows the received constellation map of the QPSK modulation under various noisy conditions for testing out the proposed AE model. The confusion matrix on the AE-based OFDM communication system with QPSK modulation is presented in Figure 4b, and results are tested at various SNR values. In the confusion matrix, the classes refer to the four QPSK symbols, each of which is equivalent to a phase shift of the carrier signal. These classes are Class 0 = [1], Class 1 = [1, −1], Class 2 = [−1, 1], and Class 3 = [−1, −1]. For the main diagonal (340,000, 305,000, 310,000, 320,000), these values belong to the respective classes. It means specific statistics include in the evaluation of a model like count or importance of the classes. The off-diagonal elements have meanings related to interaction or error between classes and are equal to 5000 in the position (1, 3). A value of 5000 in position (1, 3) means that the symbols of class 1 were wrongly labeled as class 3, 5000 times.

Figure 5 presents the various evaluation measures of the above classification model. In the first subplot, precision for each class is depicted, showing that Class 2 and Class 4 have perfect precision, while Class 1 and Class 3 have high precision values of 0.985 and 0.984, respectively. The second subplot illustrates recall, with Classes 2 and 4 achieving the highest recall of 1.000, and Classes 1 and 3 achieving recall values of 0.985 and 0.984. Subplot 3 outlines the F1-score, a combination of precision and recall. Classes 2 and 4 obtain the best F1-score of 1.000, while Classes 1 and 3 score 0.985 and 0.984, respectively. The fourth subplot presents a comparison of the total correctly classified instances and the Kappa statistic of the model. The accuracy is 99.2%, reflecting the percentage of accurately classified instances, and the Kappa coefficient is 0.992, indicating a high degree of agreement between the predicted and actual values, considering all probabilities of chance. Summing up the above indicators, it can be concluded that all the calculated metrics indicate high classification efficiency and justified performance of the proposed model. Table 6 below shows the values of the performance parameter for each class.

TABLE 6 | Performance parameter for each class of the proposed AE model.

Metric	Class 1	Class 2	Class 3	Class 4
Precision	0.985	1	0.984	1
Recall	0.985	1	0.984	1
F1-score	0.985	1	0.984	1
Accuracy		0.992		
Kappa		0.992		

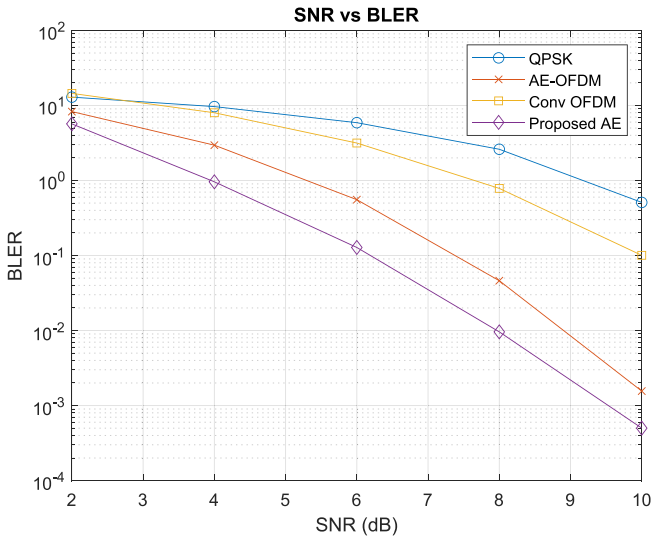

FIGURE 6 | Block error rate (BLER) versus signal-to-noise ratio (SNR) comparison for various models.

Figure 6 displays the block error rate (BLER) as a function of SNR for four modulation schemes: QPSK, Conv OFDM, AE-OFDM, and Proposed AE. A BLER value of 10−3 is assumed when one error occurs in a thousand blocks, which is a typical measure for block error performance. The results indicate that for a BLER of 10−3, the QPSK signal requires an SNR of more than 10 dB, resulting in higher error rates. Conversely, Conv OFDM achieves a BLER of 10−3 at around 8.2 dB, showing

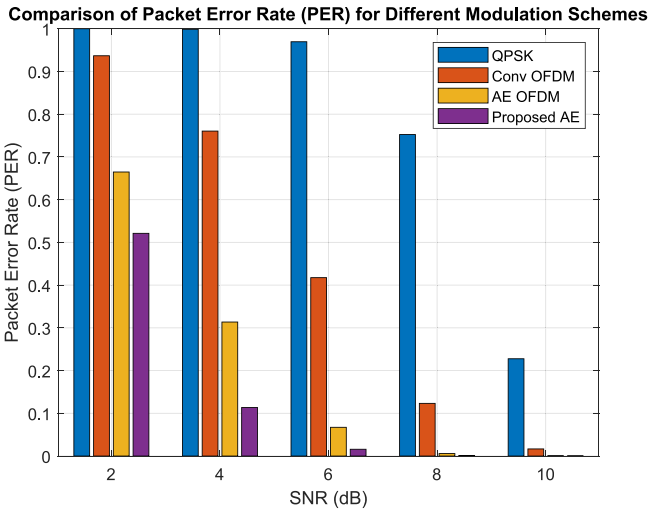

FIGURE 7 | PER comparison of proposed system.

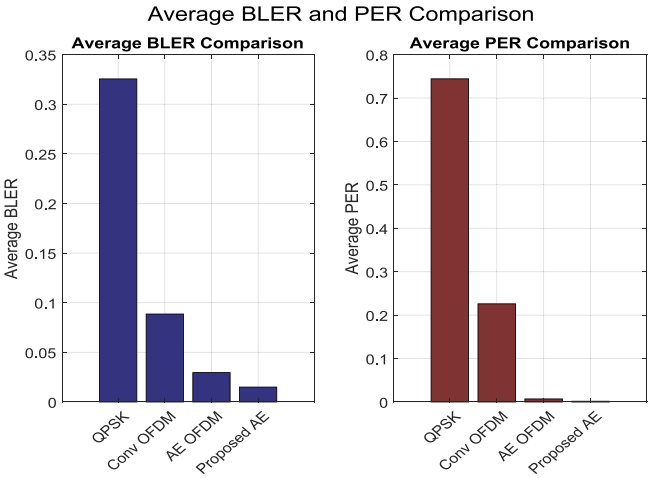

FIGURE 8 | Average BLER and average PER comparison.

some improvement but still requiring a relatively high SNR. Based on these results, AE-OFDM reaches the same BLER at approximately 5.5 dB, indicating better error performance and a lower required SNR. The Proposed AE scheme outperforms all other schemes, achieving the same BLER at only 4.2 dB, demonstrating a higher capability for reducing block errors at lower SNR levels.

TABLE 7 | Average BLER and average PER performance of various models.

Modulation scheme	Average BLER	Average PER
QPSK	0.3255	0.7442
Conv OFDM	0.0886	0.226
AE OFDM	0.0296	0.007
Proposed AE	0.015	0.0017

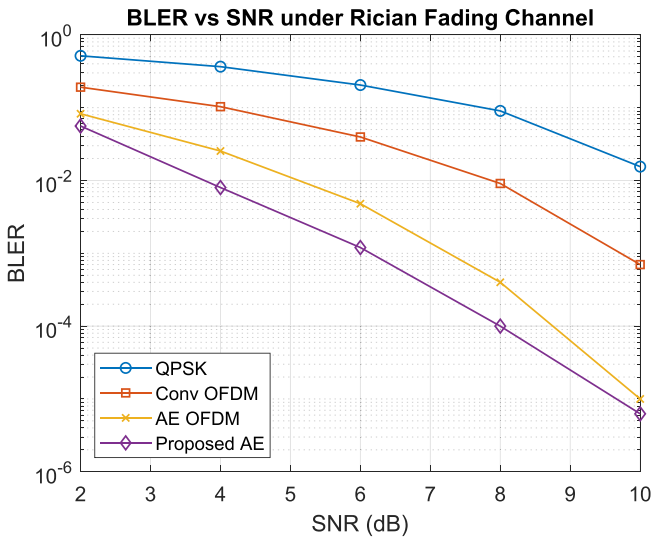

FIGURE 9 | Block error rate (BLER) versus signal-to-noise ratio (SNR) comparison for various models under Riccian model.

Figure 7 shows the comparison of PER versus SNR and highlights the performance of four modulation schemes: QPSK, Conv OFDM, AE OFDM, and Proposed AE. The data shows that QPSK exhibits the highest average PER at 0.7442, indicating the poorest performance. Conv OFDM demonstrates a significant improvement with an average PER of 0.226. AE OFDM further reduces the PER to 0.007, showcasing its effective error correction capabilities. The Proposed AE scheme achieves the lowest average PER of 0.0017, highlighting its superior performance in minimizing packet errors. Figure 8 compares the average BLER and PER also displayed by Table 7, for QPSK, Conv OFDM, AE OFDM, and Proposed AE modulation schemes. The left subplot shows that QPSK has the highest average BLER at 0.3255, indicating a high frequency of block errors. Conv OFDM improves this with an average BLER of 0.0886, while AE OFDM further reduces the BLER to 0.0296. The Proposed AE scheme achieves the lowest BLER of 0.015, demonstrating the best performance in block error reduction. The right subplot reveals a similar trend for average PER: QPSK has the highest average PER of 0.7442, Conv OFDM shows moderate performance at 0.226, AE OFDM improves to 0.007, and the Proposed AE scheme excels with the lowest average PER of 0.0017. These results collectively illustrate that the Proposed AE modulation scheme offers the best performance in both BLER and PER.

For four modulation schemes—QPSK, Conv OFDM, AE OFDM, and Proposed AE—the BLER vs SNR is shown in Figure 9 for a Rician fading channel. The graphic illustrates how error performance improves while switching from conventional to sophisticated modulation systems. The greatest BLER is shown by QPSK, which starts at 0.5153 at 2 dB SNR and drops to 0.0155 at 10 dB. With BLER values ranging from 0.1912 at 2 dB to 0.0007 at 10 dB,

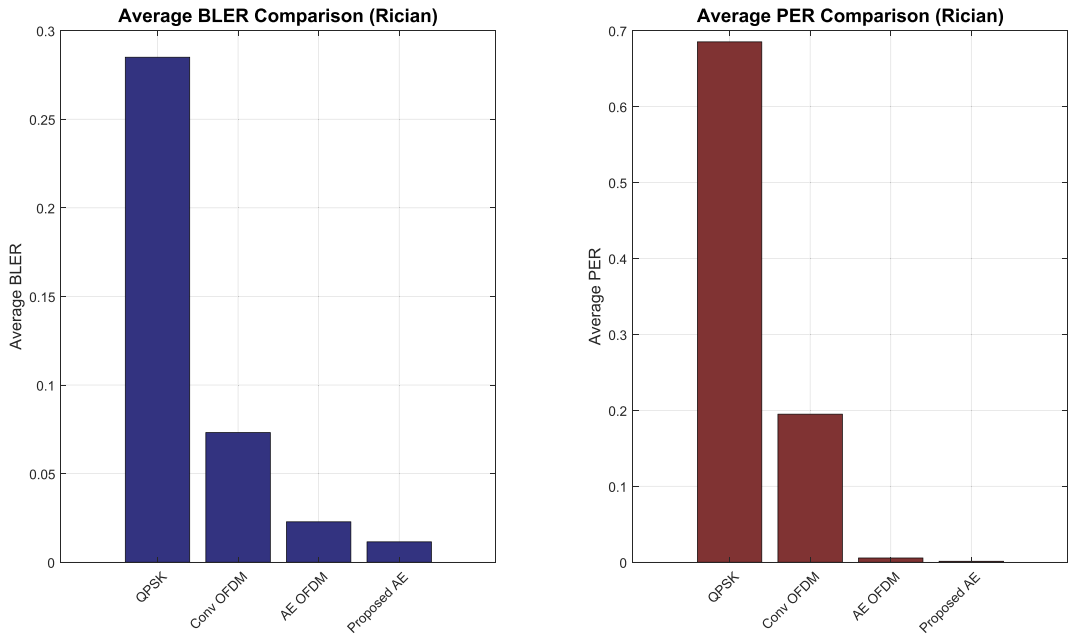

FIGURE 10 | Average BLER and Average PER comparison under Riccian model.

Conv OFDM performs better. Starting at 2 dB, the BLER is further reduced by AE OFDM, going from 0.0825 to 0.00001 at 10 dB. With BLER values ranging from 0.0560 at 2 dB to 0.00000625 at 10 dB, the Proposed AE scheme consistently outperforms the other schemes at all SNR levels. This demonstrates how well the suggested AE works to reduce error rates in Rician fading scenarios.

Figure 10 shows a comparison of the Average BLER and Average PER for four modulation schemes under a Rician fading channel: QPSK, Conv OFDM, AE OFDM, and Proposed AE. The Proposed AE (0.0115) has the lowest BLER in the Average BLER plot, whereas QPSK has the greatest BLER at 0.2850, followed by Conv OFDM (0.0732) and AE OFDM (0.0228). The suggested scheme performs better in terms of both BLER and PER under Rician fading, as seen by the Average PER plot, where QPSK has the greatest PER of 0.6852, Conv OFDM has 0.1950, AE OFDM displays 0.0056, and the Proposed AE obtains the lowest PER of 0.0013.

5 | Conclusion and Future Scope

This research introduced an improved LSTM-based AE designed to enhance the performance of OFDM systems. The framework processes one-hot encoded OFDM symbols and employs LSTM networks to reconstruct the symbols from their encoded forms. Simulations demonstrated the superiority of the proposed model over conventional OFDM systems, particularly in minimizing the Block Error Rate (BLER) and PER. For example, the Average BLER for the Proposed AE was reduced to 0.0150, compared with 0.0296 for the conventional AE and 0.0886 for convolutional OFDM. Similarly, the Average PER dropped to 0.0017, a significant improvement over 0.0070 for the conventional AE and 0.2260 for convolutional OFDM.

While the results are promising, the real-world implementation of the proposed LSTM-based AE presents challenges that require further investigation. These include addressing computational complexity for deployment in resource-constrained devices, ensuring low-latency processing for real-time communication, and optimizing the model's robustness to handle dynamic channel conditions and interference in diverse environments. Additionally, integrating the proposed model into existing communication infrastructure may require addressing compatibility and scalability issues. Future work will focus on mitigating these challenges by exploring lightweight model architectures, efficient hardware implementations, and advanced training techniques tailored to real-world deployment. This research lays the groundwork for the development of resilient and high-performance communication systems, paving the way for practical applications in next-generation wireless networks.

Acknowledgments

The authors have nothing to report.

Ethics Statement

This article does not contain any studies with human participants performed by any of the authors.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

Research data are not shared.

References

1. M. Meenalakshmi, S. Chaturvedi, and V. K. Dwivedi, "Deep Learning Techniques for OFDM Systems," *IETE Journal of Research* 69, no. 9 (2023): 5883–5897.

2. B. Jdid, K. Hassan, I. Dayoub, W. H. Lim, and M. Mokayef, "Machine Learning Based Automatic Modulation Recognition for Wireless Communications: A Comprehensive Survey," *IEEE Access* 9 (2021): 57851–57873.

3. S. A. Mohsan, Y. L. Hassnain, A. V. Shvetsov, J. Varela-Aldás, S. M. Mostafa, and A. Elfikky, "A Survey of Deep Learning Based NOMA: State of the Art, Key Aspects, Open Challenges and Future Trends," *Sensors* 23, no. 6 (2023): 2946.

4. T. Toma and T. Wada, A Study of OFDM Communication System With Autoencoder, Technical Report of the Institute of Electronics, Information and Communication Engineers, 2022.5, vol 122, no. SR-12, SR2022-6, pp. 27–33.

5. T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking* 3, no. 4 (2017): 563–575, https://doi.org/10.1109/TCCN.2017.2758370.

6. S. S. Darly, D. Kadhiravan, K. Hemachandran, and M. Rege, "Simulation Strategies for Analyzing of Data," in *Handbook of Artificial Intelligence and Wearables*, 1st Edition (CRC Press, 2024), 27–64, https://doi.org/10.1201/9781032686714-3.

7. J. Masci, U. Meier, D. Cireşan, and J. Schmidhuber, "Stacked Convolutional Auto-Encoders for Hierarchical Feature Extraction," in *Artificial Neural Networks and Machine Learning – ICANN 2011*, ed. T. Honkela, W. Duch, M. Girolami and S. Kaski(Springer, 2011), 52–59.

8. A. Jayamathi and T. Jayasankar, "Deep Learning Based Stacked Sparse Autoencoder for PAPR Reduction in OFDM Systems," *Intelligent Automation & Soft Computing* 31, no. 1 (2022): 311–324.

9. L. L. Hao, C. D. Li, and D. Y. Wang, "Quantitative Analysis of the Stacked Autoencoder Method in MIMO-ACO-OFDM VLC Systems," *Lighting Research & Technology* 53, no. 1 (2021): 54–73.

10. Y. Huleihel and H. H. Permuter, "Low PAPR MIMO-OFDM Design Based on Convolutional Autoencoder," *IEEE Transactions on Communications* 72, no. 5 (2024): 2779–2792, https://doi.org/10.1109/TCOMM.2023.3348839.

11. T. Wada, T. Toma, M. Dawodi, and J. Baktash, "A Denoising Autoencoder Based Wireless Channel Transfer Function Estimator for OFDM Communication System," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)* (IEEE, 2019), 530–533.

12. A. Ali, F. Yangyu, and S. Liu, "Automatic Modulation Classification of Digital Modulation Signals With Stacked Autoencoders," *Digital Signal Processing* 71 (2017): 108–116.

13. A. Dai, H. Zhang, and H. Sun, "Automatic Modulation Classification Using Stacked Sparse Auto-Encoders," in *2016 IEEE 13th International Conference on Signal Processing (ICSP)* (IEEE, 2016), 248–252.

14. S. Duan, K. Chen, Y. Xiang, and M. Qian, "Automatic Multicarrier Waveform Classification via PCA and Convolutional Neural Networks," *IEEE Access* 6 (2018): 51365–51373.

15. I. Kalphana and T. Kesavamurthy, "Convolutional Neural Network Auto Encoder Channel Estimation Algorithm in MIMO-OFDM System," *Computer Systems Science and Engineering* 41, no. 1 (2022): 171–185.

16. B. Lin, X. Wang, W. Yuan, and W. Nan, "A Novel OFDM Autoencoder Featuring CNN-Based Channel Estimation for Internet of Vessels," *IEEE Internet of Things Journal* 7, no. 8 (2020): 7601–7611.

17. S. Tsugawa, T. Toma, S. Oshiro, and T. Wada, "Scalability in Autoencoder-Based OFDM Communication System," in *2023 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI)* (India: Gwalior, 2023), 1–5, https://doi.org/10.1109/CVMI59935.2023.10464866.

18. A. Mohamed, A. S. Tag Eldien, M. M. Fouda, and R. S. Saad, "LSTM-Autoencoder Deep Learning Technique for PAPR Reduction in Visible Light Communication," *IEEE Access* 10 (2022): 113028–113034, https://doi.org/10.1109/ACCESS.2022.3216574.

19. Z. Lin, X. Hu, Y. Zhang, et al., Splitlora: A Split Parameter-Efficient Fine-Tuning Framework for Large Language Models. *arXiv preprint arXiv:2407.00952* 2024.

20. R. Liang, J. Chen, K. He, et al., Ponziguard: Detecting Ponzi Schemes on Ethereum With Contract Runtime Behavior Graph (crbg). In *Proceedings of the 46th IEEE/ACM International Conference on Software Engineering*, pp. 1–12. 2024.

21. X. Yang, J. Chen, K. He, H. Bai, W. Cong, and D. Ruiying, "Efficient Privacy-Preserving Inference Outsourcing for Convolutional Neural Networks," *IEEE Transactions on Information Forensics and Security* 18 (2023): 4815–4829.

22. P. Vishwakarma and K. Ahuja, "A Review on Machine Learning Assisted Handover Mechanisms for Future Generation Wireless Networks," *International Journal of Scientific Research & Engineering Trends* 10, no. 5 (2024): 255–269, https://doi.org/10.61137/ijsret.vol.10.issue5.255.

23. B. S. d. C. da Silva, V. D. P. Souto, R. D. Souza, and L. L. Mendes, "A Survey of PAPR Techniques Based on Machine Learning," *Sensors* 24, no. 6 (2024): 1918.

24. M. E. Ali and A. A. Abdulkafi, "OFDM-Based VLC Systems: A Systematic Review," in *BIO web of Conferences*, vol. 97 (EDP Sciences, 2024), 00090.

25. P. Lohan, B. Kantarci, M. A. Ferrag, N. Tihanyi, and Y. Shi, "From 5G to 6G Networks, a Survey on AI-Based Jamming and Interference Detection and Mitigation," *IEEE Open Journal of the Communications Society* 5 (2024): 3920–3974, https://doi.org/10.1109/OJCOMS.2024.3416808.

26. C. Mao, M. Zongwen, Q. Liang, I. Schizas, and C. Pan, "Deep Learning in Physical Layer Communications: Evolution and Prospects in 5G and 6G Networks," *IET Communications* 17, no. 16 (2023): 1863–1876.

27. X. Zhang, Z. Luo, W. Xiao, and L. Feng, "Deep Learning-Based Modulation Recognition for MIMO Systems: Fundamental, Methods, Challenges," *IEEE Access* 12 (2024): 112558–112575, https://doi.org/10.1109/ACCESS.2024.3440350.

28. Z. Lin, G. Zhu, Y. Deng, et al., "Efficient Parallel Split Learning Over Resource-Constrained Wireless Edge Networks," *IEEE Transactions on Mobile Computing* 23, no. 10 (2024): 9224–9239, https://doi.org/10.1109/TMC.2024.3359040.

29. Z. Lin, G. Qu, W. Wei, X. Chen, and K. K. Leung. Adaptsfl: Adaptive Split Federated Learning in Resource-Constrained Edge Networks. *arXiv preprint arXiv:2403.13101* 2024.

30. Z. Lin, Z. Chen, Z. Fang, X. Chen, X. Wang, and Y. Gao. Fedsn: A General Federated Learning Framework Over Leo Satellite Networks. *arXiv preprint arXiv:2311.01483* 2023.

31. Y. Zhang, Z. Lin, Z. Chen, Z. Fang, W. Zhu, X. Chen, J. Zhao, and Y. Gao. Satfed: A Resource-Efficient Leo Satellite-Assisted Heterogeneous Federated Learning Framework. *arXiv preprint arXiv:2409.13503* 2024.

32. R. Liang, J. Chen, C. Wu, et al., Vulseye: Detect Smart Contract Vulnerabilities via Stateful Directed Graybox Fuzzing. *arXiv preprint arXiv:2408.10116* 2024.