Advances in Artificial Intelligence and Machine Learning; Research 5 (1) 3548-3564 Received 21-12-2024; Accepted 22-03-2025; Published 29-03-2025

Next-Generation Noise-Resilient Communication Receiver Design Using Denoising Autoencoders

Varsha PS varshaps@ceconline.edu
Department of Electronics
College Of Engineering Chengannur,IHRD
APJ Abdul Kalam Technological University
Kerala,India

Hari VS harivs@ceconline.edu
Department of Electronics
College Of Engineering Chengannur,IHRD
APJ Abdul Kalam Technological University
Kerala,India

Corresponding Author: Varsha PS

Abstract

Autoencoders have become a prominent focus in unsupervised learning research due to their ability to capture essential data features, perform efficient dimensionality reduction and aid in noise reduction. In this paper, we propose a novel approach that integrates autoencoders with deep learning classifiers for the efficient reception of Binary Phase Shift Keying signals. Specifically, three distinct autoencoders–Linear, Long Short-Term Memory, and Convolutional–are cascaded with deep learning classifiers to denoise received signals corrupted by Additive White Gaussian Noise. This streamlined methodology not only enhances signal quality and interpretability but also facilitates the development of a more efficient receiver, outperforming conventional designs that rely on multiple processing blocks paving the way for robust and adaptive communication systems.

Keywords: Autoencoders, BPSK, Linear autoencoder, LSTM autoencoder, Convolutional autoencoder, Deep learning classifiers, MSE,BER

1. INTRODUCTION

High-speed data transport across several platforms is made possible by digital communication networks, which are the foundation of contemporary information exchange [1]. In hostile situations, communication lines are frequently vulnerable to noise and interference, compromising reliable detection and degrading signal quality [2]. Consequently, receivers adept at managing the diverse impairments caused by noise and interference in the communication channel are essential for deciphering incoming signals and accurately recovering transmitted data [3–5]. The correlation receiver [6], is a crucial design among the several receiver types utilised in digital communication. Owing

Citation: Varsha PS and Hari VS. Next-Generation Noise-Resilient Communication Receiver Design Using Denoising Autoencoders. Advances in Artificial Intelligence and Machine Learning. 2025;5(1):203.

to its user-friendliness, affordability, and efficacy in enhancing Signal-to-Noise Ratio (SNR), it has been utilised for an extended period in systems reliant on binary modulation techniques, including Binary Phase Shift Keying (BPSK) and Quadrature Phase Shift Keying (QPSK). Conversely, these designs are most effective in the presence of consistent, predictable noise. However, they contend with additive white Gaussian noise and interference since they are designed for idealised settings and lack the adaptive, nonlinear capabilities required to mitigate complex noise and interference. Their inability to adapt to varying noise levels, combined with threshold limitations and linear processing, results in an elevated Bit Error Rate in loud or challenging conditions. Noise and interference can collect in pulse-shaped signals over multiple bit intervals, especially when sent over extended distances or through memory-containing channels. These receivers may neglect the cumulative impact of noise over successive bits, as they evaluate signals on a per-symbol basis, potentially leading to decoding issues.

Recent developments in deep learning(DL) [7–11], have shown immense potential in getting over the limitation of traditional receivers. Techniques such as Long short-Term Memory (LSTM), Convolutional Neural Networks (CNN), and more recently, Transformer architectures [12–14], have demonstrated exceptional flexibility in handling a variety of unpredictable noise environments, outperforming conventional receivers in terms of bit error rate (BER) and signal recovery in the presence of additive white Gaussian noise. Numerous studies have laid the groundwork for integrating neural networks into communication systems, achieving superior performance in the face of complex noise environments. In their early approach for using CNNs in modulation and signal decoding tasks, O'Shea and Hoydis (2017) [15], introduced the idea of end-to-end learning for the physical layer. In a similar vein, Ye, Li, and Juang (2018) [16], demonstrated notable improvements in noise tolerance by utilising CNNs and LSTM networks for channel estimation and detection in OFDM systems. Samuel et al.(2017) [17], expanded on these ideas with CNN and LSTM models for MIMO detection and time-varying channel adaptation, respectively, which underscored the ability of deep learning to dynamically adjust to channel conditions. Huang et al. (2019) [18], further explored the role of LSTMs and CNNs in managing 5G environments, highlighting these models' adaptability to complex, multi-path channels.Other contemporary studies also demonstrate the long-term resilience of these deep learning models, which can dynamically adapt to the varying characteristics of communication channels without depending on predetermined assumptions about noise characteristics by utilizing data-driven features. In our previously published work, Intelligent Deep Learning-Based Speech Receivers [19], we explored how deep learning techniques can effectively decipher transmitted data in the presence of AWGN, demonstrating superior noise mitigation capabilities compared to conventional correlation receivers.

We propose the integration of autoencoders as a significant improvement in receiver design, building on these advancements. To enhance receiver design, autoencoders are used as a preprocessing layer integrated with deep learning models, resulting in improved BER performance and enhanced noise resilience. Our model utilises LSTM, convolutional, and linear autoencoders [20, 21], to adapt to different noise profiles, efficiently denoising signals prior to final processing by the deep learning layer. LSTM-based autoencoders effectively manage sequential dependencies, whereas convolutional autoencoders excel at capturing spatial noise patterns, demonstrating particular efficacy in high-SNR and AWGN-dominated environments. This cascaded architecture effectively decreases BER through dynamic adaptation to complex noise, thereby exceeding the performance of conventional and standard deep learning receivers, and providing substantial, data-driven improvements in noisy communication environments.

2. BACKGROUND AND THEORETICAL FRAMEWORK

2.1 Binary Phase Shift Keying (BPSK) Conventional Transmission System

A sequence of N input bits, b_i is produced during BPSK transmission, where each b_i is either 0 or 1. NRZ (Non-Return-to-Zero) modulation [22], is then used to transfer each bit b_i to a symbol s_i with amplitude values of -A and +A. A pulse shaping filter, typically a Raised Cosine pulse(RCC) $p(t)$ [23], shapes the modulated signal to create a continuous-time signal $x(t)$ in order to increase spectral efficiency and reduce inter-symbol interference (ISI).

$$x(t) = \sum_{i=0}^{N-1} s_i p(t - iT) \tag{1}$$

where T is the symbol duration.

Additive White Gaussian Noise $n(t)$, that features a noise power spectral density (N_0) and uses noise samples taken from a Gaussian distribution with zero mean and variance (σ^2), corrupts the signal ($x(t)$) during transmission:

$$n(t) \sim N(0, \sigma^2) \tag{2}$$

The received signal $r(t)$ at the receiver end is the sum of the transmitted signal $x(t)$ and noise $n(t)$:

$$r(t) = x(t) + n(t) \tag{3}$$

To assess BER performance under varied noise situations, the noise variance σ^2 is changed based on different Signal to Noise Ration(SNR) levels. The correlation receiver determines the correlation between the pulse shape $p(t)$ and the received signal segment $r(t)$ in order to detect each sent bit b_i. This decision rule can be expressed as:

$$\hat{b}_i = \begin{cases} 1, & if \quad \sum(r_i \cdot p(t)) > 0, \\ -1, & \text{otherwise.} \end{cases} \tag{4}$$

where r_i is the received signal corresponding to the bit. The BER for each SNR level is computed by comparing the detected bits $\hat{b_i}$ with the original transmitted bits b_i:

$$BER = \frac{\sum_{i=0}^{N-1} \mid \hat{b_i} - b_i \mid}{N} \tag{5}$$

The theoretical BER for BPSK in an AWGN channel is given by:

$$Theoretical\ BER = \frac{1}{2} erfc(sqrt(SNR)) \tag{6}$$

By emphasising on how closely the received signal resembles the pulse shape, this correlation-based detection improves robustness and offers resilience against noise.

2.2 Deep Learning Models as Advanced Alternatives for Receiver Design

Previous research [19, 24], of ours investigated how sophisticated deep learning architectures [10], could be applied at the receiver level to improve the accuracy and robustness of signal decoding in

the presence of communication noise. To address the issue of additive white Gaussian noise, we developed LSTM-based receiver models that are able to effectively reduce its impact by capitalising on the temporal linkages present in sequential data. The memory persistence and adaptive learning capabilities of these models were enhanced by sequential LSTM layers, which were reinforced by thick layers that utilised regularisation techniques like dropout to prevent overfitting. Bit error rate and other performance indicators were substantially improved when the ADAM optimiser and binary cross-entropy [24–26], loss were used together, laying the groundwork for practical application.

The effectiveness of CNN receiver models in extracting and using spatial aspects of noised signals was further proven by our work with Conv1D and Conv2D variations. As a result of utilising two-dimensional data transformations, the Conv2D model provided a more thorough feature representation than the Conv1D model, which was optimised for one-dimensional sequential data. Combining sophisticated optimisation techniques with the layered architectures of convolutional, pooling, and dense layers, these models demonstrated their efficacy in precise binary classification tasks. These earlier studies highlight the possibility of reimagining conventional paradigms of signal decoding by combining LSTM and CNN designs at the receiver level.

The architecture and learning process of the LSTM and CNN models–how they convert information into output probabilities–form the basis of their prediction process. For both LSTM and CNN models:

$$\hat{b} = \begin{cases} 1, \text{ if } P(r = 1 \mid X; \theta) \geq 0.5, \\ 0, \text{ if } P(r = 1 \mid X; \theta) < 0.5. \end{cases} \tag{7}$$

- $P(r = 1 \mid X; \theta)$ is the probability that the received bit is 1, computed by the model.
- $X = [x1, x2, ..., xP]$ represents the input feature vector of length P.
- θ represents the learned parameters (weights, biases) of the model.

The input features (X) are processed by the LSTM or CNN through their levels to calculate $P(r = 1 \mid X; \theta)$, which is usually the result of the sigmoid activation in the last layer.

2.3 Autoencoder Architectures

Effective data representations can be learnt unsupervisedly using autoencoders [26–28], a kind of artificial neural network.These models are composed of two components. An encoder that compresses the input data into a latent-space representation and a decoder that reconstructs the input from this compressed form as shown in FIGURE 1. An autoencoder's goal is to reduce errors in reconstruction between the input and the output. They are quite adaptable and frequently used for noise reduction, feature extraction, and data compression. This work explores three types of autoencoders: Linear, LSTM and Convolutional-based. Each architecture contributes uniquely to noise removal, highlighting the versatility of autoencoders in communication and signal processing applications.

A linear autoencoder [28, 29] uses fully linked linear layers for both the encoder and the decoder. It does not have the advanced representation capabilities of non- linear models, but it is nevertheless

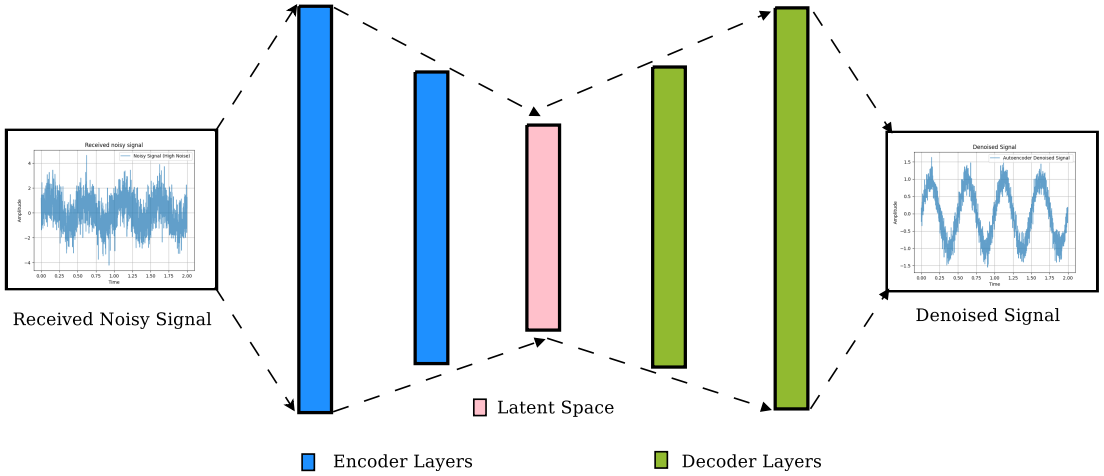

Figure 1: Architecture of an Autoencoder for Signal Denoising

a good method for eliminating noise from structured, low-dimensional data. A linear autoencoder carefully transforms the noisy input signal into a clean, reconstructed output for noise reduction. Initially, the autoencoder receives the input noisy signal . By reducing the dimensionality of the input, the encoder effectively filters away noisy components while maintaining the essential features. The Latent Space, a compressed representation that captures the key features of the input signal without the majority of noise distortions, is the end result of this transformation. The decoding stage then re- constructs the signal from this latent representation to reverse the compression process and maintain the noise-free properties that were learnt during encoding. Ultimately, a denoised version of the original input emerges as the output clean signal, ready for further processing. This procedure demonstrates in detail how autoencoders extract and preserve significant data while eliminating noise by leveraging their compression and reconstruction capabilities.

Advanced architectures such as LSTM and Convolutional autoencoders [20, 30–32], which build on the foundation of linear autoencoders, significantly improve noise removal in BPSK transmission over an AWGN channel. These architectures make use of spatial patterns, temporal dependencies, and non-linear relationships to improve signal reconstruction and mitigate noise. In order to effectively capture long-term temporal dependencies, the LSTM autoencoder architecture uses LSTM layers in both the encoder and the decoder. The encoder compresses sequential data into a compact latent space, and the decoder uses the learnt latent representation to reconstruct the original signal. LSTM layers are well-suited for time-series data because they can handle varying noise levels across sequential bit streams, such as those encountered in BPSK signals. The symmetric LSTM layers in the encoder and decoder guarantee that the temporal features are efficiently compressed and reconstructed. Convolutional layers constitute the basis for encoding in convolutional autoencoder architecture, and transposed convolutional layers are employed for decoding. Because convolutional layers are good at extracting spatial features, they are perfect for working with signals that are displayed in two dimensions, like spectrograms or reshaped signal features. The isolation of significant patterns in spatial data and extraction using convolutional layers filters noise. In the latent space, these characteristics are further improved, reducing the influence of AWGN. Thus by focusing on key signal characteristics and adapting to varying noise patterns, these models

demonstrate significant improvements in signal reconstruction, particularly in noisy communication environments.

2.4 Detailed System Workflow

The proposed method improves communication reliability in noisy conditions by employing sophisticated signal processing techniques. By combining autoencoders with deep learning classifiers, the system denoises signals corrupted by an AWGN channel efficiently when compared decoding with deep learning classifiers alone. The block diagram FIGURE 2, explains the complete flow of a dependable communication system, which effectively reduces noise in BPSK transmissions through the integration of autoencoders and deep learning models. The procedures include generating binary data, modulating it with BPSK, and simulating its transmission via an AWGN channel using a Python-based implementation.

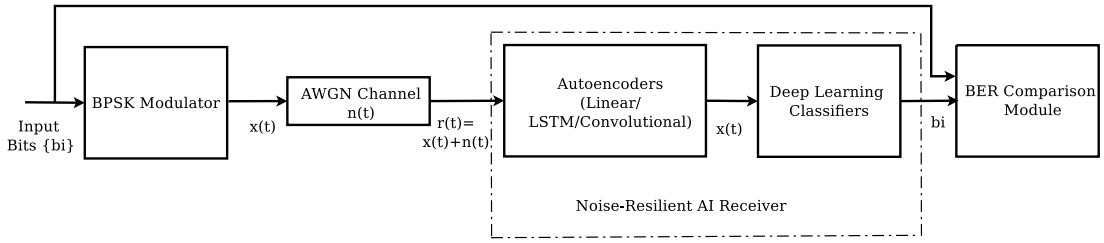

Figure 2: System Workflow for Noise Mitigation in BPSK Transmission Using Autoencoders and Deep Learning Classifiers

2.4.1 Transmitter

The system initiates by generating a stream of random binary bit sequences, b_i (0's and 1's) that require transmission over a noisy channel. This data represents the digital information that will undergo modulation and subsequent processing. We modulated the binary data using Non-Return-to-Zero (NRZ) Binary Phase Shift Keying (BPSK). This modulation method makes sure that binary data $\{0, 1\}$ are mapped to the transmitted signal with two possible amplitudes $\{-1, 1\}$. This way, the signal stays the same during the bit duration, which lowers the chance of errors and makes the communication link more reliable in a range of transmission environments.The NRZ signal is processed through a Root Raised Cosine filter $h_r(t)$ to shape the waveform, ensuring smooth transitions, minimizing inter-symbol interference (ISI), and optimizing bandwidth utilization. During receiver training, various roll-off factor values ($0 < \alpha < 1$) are considered to evaluate their impact on system performance. Additionally, the impact of up-sampling on the signal power relative to noise SNR is considered. Since up-sampling to 49 samples per bit can introduce an apparent improvement due to increased signal power, care is taken to ensure that the observed performance gains arise from the deep learning model's effectiveness rather than an artificial boost in SNR. This is done by ensuring proper noise generation and normalization by confirming that noise power is correctly scaled relative to signal power (accounting for up-sampling) prevents artificial improvements in

SNR.

$$h_r(t) = \begin{cases} \frac{4\alpha}{\pi\sqrt{T}} \cdot \frac{\cos((1+\alpha)\pi t/T) + \frac{T}{4\alpha t}\sin((1-\alpha)\pi t/T)}{1-\left(\frac{4\alpha t}{T}\right)^2}, & \text{if } t \neq \pm\frac{T}{4\alpha}, \\ \frac{\alpha}{\sqrt{2T}}\left[\left(1+\frac{2}{\pi}\right)\sin\left(\frac{\pi}{4\alpha}\right) + \left(1-\frac{2}{\pi}\right)\cos\left(\frac{\pi}{4\alpha}\right)\right], & \text{if } t = \pm\frac{T}{4\alpha}, \\ \frac{1-\alpha+\frac{4\alpha}{\pi}}{\sqrt{T}}, & \text{if } t = 0. \end{cases} \tag{8}$$

Thus a modulated signal, $x(t)$ is ready for transmission through the channel. The Additive White Gaussian Noise channel block is an essential component in communication systems.It introduces disturbances in the transmitted signal, emulating the random noise, $n(t)$ that affects real communication channels, such as thermal noise or electromagnetic interference.The received signal, $r(t)$ is a sum of the transmitted signal, $x(t)$ and noise, $n(t)$.

$$r(t) = x(t) + n(t) \tag{9}$$

2.4.2 Receiver

The receiver is designed to handle noisy signal, $r(t)$ transmitted through an Additive White Gaussian Noise channel. It leverages a two-stage approach:

- Denoising Autoencoders (DAE)–Linear, LSTM, or Convolutional– that mitigates noise by reconstructing clean signals, $\hat{x}(t)$ from noisy inputs.
- Deep Learning Classifier which processes the denoised signal to predict binary values, $\hat{b}_i$ (0 or 1).

By utilising deep learning capabilities, autoencoders offer a contemporary, data-driven alternative for the traditional correlation-based digital communication receiver architecture. The linear processing nature of correlation receivers, their incapacity to dynamically adjust to fluctuating noise levels, and their disdain for cumulative noise effects across subsequent bits are some of their drawbacks. Conversely, autoencoders are unsupervised neural networks that are made to efficiently learn input data representations while maintaining key characteristics. They use their compression and reconstruction capabilities to their advantage when it comes to noise reduction. The receiver dynamically learns and adjusts to the statistical characteristics of the received noisy signals by substituting an autoencoder for the correlation process. This offers a number of benefits, including enhanced BER, dynamic noise reduction, non-linear processing, and feature extraction.

2.4.3 Signal sampling and processing

The transmitted signal is inherently continuous, whereas deep learning models operate on discrete-time representations. To bridge this gap, a systematic sampling process is employed to convert into a discrete form suitable for processing by deep learning architectures. The choice of sampling rate plays a crucial role in ensuring that sufficient signal information is retained while avoiding aliasing effects. In this study, the continuous signal is sampled at a predefined rate of N=49 to generate

49 discrete samples per symbol. This sampling rate is selected to balance computational efficiency and accuracy, ensuring that the essential features of the transmitted waveform are preserved. The Raised cosine pulse shaping function used for modulation influences the spectral and temporal characteristics of the sampled signal.

For the deep learning-based equalization methods, different sampling and transformation techniques are applied depending on the autoencoder architecture:

- Linear Autoencoder and LSTM Autoencoder: The sampled waveform is processed sequentially with 49 samples per bit used as input to the models. These models operate on one-dimensional time-series representations of the received signal.
- CNN Autoencoder: The 49 samples per bit are reshaped into a 7 X 7 2D representation before being fed into the convolutional neural network. This transformation allows the CNN to extract spatial features from the received signal.

In this study, we assume ideal symbol timing and carrier phase synchronization to isolate the impact of channel equalization techniques on the BER performance. This assumption is commonly used in deep learning-based equalization studies to focus on signal reconstruction without additional distortions from synchronization errors.Future work could explore integrating synchronization mechanisms within the deep learning model itself, allowing for joint equalization and synchronization optimization.

2.4.4 Model architectures, training and testing process

Data is generated by simulating noisy signals using the AWGN channel model:

- Noise variance, σ^2 is varied to achieve SNR levels from -20 dB to 20 dB.
- Original transmitted bits, b_i are mapped to modulated signals, $x(t)$ using NRZ-BPSK modulation.
- Noisy signals, $r(t)$ are obtained by adding Gaussian noise, $n(t)$ to $x(t)$.

A non-overlapping sliding window of N' samples is taken into consideration for each sequentially received signal, $r(t)$. These samples correspond to a single transmitted bit. N' features(with noise) per bit are taken as input to our model. The sliding window extracts features from the received signal, forming the input for the autoencoders. These features capture the characteristics of the received noisy signal. The received noisy data and the clean signal(N' features without noise per bit)are stored in a CSV file. The clean signal serves as the target for supervised learning, enabling the model to learn the mapping from noisy to clean signals. Data is normalized using Min-Max Scaling to scale features to improve the stability and convergence of training, ensuring the model handles features effectively. To address the challenges of mitigating additive white Gaussian noise in communication signals, three distinct autoencoder architectures were explored are Linear Autoencoder, LSTM Autoencoder, and Convolutional Autoencoder (Conv2D). Every

architecture is customised to fit certain noise profiles and data properties. Because of its computational efficiency and flat vector input architecture, the Linear Autoencoder is appropriate for low-dimensional structured data. Time-series signals benefit from the LSTM Autoencoder's capacity to capture temporal dependencies, which makes it highly effective in processing sequential data. The Convolutional Autoencoder shows its efficacy in handling image-like or spatial data representations by using convolutional layers to extract spatial patterns. The TABLE 1, 2, 3 below summarizes the architectural components like encoder,latent and decoder layers, input-output formats, activation functions and strenghts of these three autoencoder types.

Table 1: Linear Autoencoder Architecture

Feature	**Description**
Input	Flat vector of size *input_size* (N'=49)Noisy signal
Encoder Layers	Fully connected (Linear) layers: - Layer 1: *input_size* → 128 - Layer 2: 128 → 64 - Layer 3: 64 → 12
Latent Space	12-dimensional vector
Decoder Layers	Fully connected (Linear) layers: - Layer 1: 12 → 64 - Layer 2: 64 → 128 - Layer 3: 128 → *input_size*
Activation Functions	ReLU, Sigmoid
Output	Flat vector of size *input_size*(N"=49) Denoised Signal
Strengths	Simple and computationally efficient

Table 2: LSTM Autoencoder Architecture

Feature	**Description**
Input	Sequence of vectors (time steps × input features=1× N'=49)
Encoder Layers	*input_size*=49, *hidden_size*=64, *latent_size*=32 LSTM layers: - LSTM Layer 1: *input_size* → *hidden_size* - LSTM Layer 2: *hidden_size* → *hidden_size* - Fully Connected: *hidden_size* → *latent_size*
Latent Space	Latent vector of size *latent_size*
Decoder Layers	Fully connected + LSTM layers: - Fully Connected: *latent_size* → *hidden_size* - LSTM Layer 1: *hidden_size* → *hidden_size* - LSTM Layer 2: *hidden_size* → *hidden_size* - Fully Connected: *hidden_size* → *input_size*
Activation Functions	ReLU, Sigmoid
Output	Reconstructed sequence of vectors
Strengths	Handles sequential data effectively

The encoder and decoder layer symmetry ensures that the network has balanced complexity for encoding and decoding. The fully connected linear layers in the encoder part of the linear autoencoder, with progressively decreasing dimensions, assist in learning complex patterns in the noisy

Table 3: Convolutional Autoencoder (Conv2D) Architecture

Feature	Description
Input	2D array (49 features converted to 7×7 feature map for Conv2D processing)
Encoder Layers	Convolutional layers: - Conv2D Layer 1: $1 \rightarrow 128$ filters - Conv2D Layer 2: $128 \rightarrow 64$ filters - Conv2D Layer 3: $64 \rightarrow 16$ filters
Latent Space	16-channel feature map, downsampled by Conv2D
Decoder Layers	Transposed Convolutional layers: - Transposed Conv2D Layer 1: $16 \rightarrow 64$ filters - Transposed Conv2D Layer 2: $64 \rightarrow 128$ filters - Transposed Conv2D Layer 3: $128 \rightarrow 1$ filter
Activation Functions	ReLU, Sigmoid
Output	Reconstructed 2D array (7×7)
Strengths	Handles spatial data effectively

input. They compress input features to create a low-dimensional latent space, which forces the network to focus on the most salient features, effectively discarding noise. Then the progressively increasing decoder layers expand intermediate features and output the reconstructed denoised signal. When it comes to LSTM autoencoders, the combination of LSTM layers and fully connected layers enables nonlinear mapping, which is crucial for denoising tasks. The LSTM layers excel at modeling time-dependent patterns in noisy signals. In a convolutional autoencoder, the process begins with preprocessing the noisy signal into a 2D format. The CNN Autoencoder architecture consists of an encoder with three convolutional layers, using kernel sizes of 3 X 3, followed by a downsampling operation with a stride of 2. The decoder mirrors this structure using transposed convolutional layers to reconstruct the signal. The encoder employs 128, 64, and 16 channels in its three convolutional layers, respectively, while the decoder symmetrically uses 16, 64, and 128 channels before the final reconstruction layer. The number of hidden units in the fully connected layers of the Linear and LSTM Autoencoders is set to 128, 64, and 12, with a latent dimension of 12 for feature compression. The use of ReLU activation [32], in the encoder and decoder allows the network to model non-linear relationships in the data.Sigmoid activation [32], in the output ensures the reconstructed signal remains within the normalized range of the input data.

In order to balance gradient stability and computing efficiency during training, all models adopt a standard batch size. With a learning rate of 0.0001, the Adam optimiser is used by the Linear and LSTM Autoencoders, which take advantage of its adaptive gradient updates to manage a variety of data complexity. With the same optimiser, the Convolutional Autoencoder employs a little higher learning rate of 0.001, which frequently converges more quickly because of its localised feature extraction. As a reflection of their architectural requirements, each model is trained over a varying number of epochs. We adjust these hyperparameters based on the loss computation, aiming to balance convergence speed, computational overhead, and model capacity.

The objective of training is to minimize the Mean Squared Error(MSE) [27, 33], loss thus ensuring the predicted output closely resembles the clean target signal. The loss function used for all three autoencoders is MSE, which measures the average squared difference between the predicted output

and the original signal. A thorough comparison of the MSE calculation for each architecture, including the input and output shapes and the associated mathematical formula, is given in the TABLE 4.

Table 4: Comparison of Loss Calculations

Autoencoder	Input Shape	Output Shape	MSE Formula
Linear	49	49	$MSE_{Linear} = \frac{1}{49}\sum_{j=1}^{49}\left(y_j - \hat{y}_j\right)^2$ where: y_j: The j-th element of the original data. $\hat{y}_j$: The j-th element of the reconstructed data.
LSTM	1×49	1×49	$MSE_{LSTM} = \frac{1}{49}\sum_{k=1}^{49}\left(y_k - \hat{y}_k\right)^2$ where: y_k: The k-th element of the original data. $\hat{y}_k$: The k-th element of the reconstructed data.
Convolutional	$1 \times 7 \times 7$	$1 \times 7 \times 7$	$MSE_{Conv} = \frac{1}{49}\sum_{i=1}^{7}\sum_{j=1}^{7}\left(y_{i,j} - \hat{y}_{i,j}\right)^2$ where: $y_{i,j}$: The sample value at position (i, j) in the original data. $\hat{y}_{i,j}$: The sample value at position (i, j) in the reconstructed data.

2.4.5 Testing and prediction pipeline

During the testing phase, as the sequential received signal, $r(t)$ arrives, it is first processed by the trained autoencoders. Every 49-sample segment from $r(t)$ is extracted out and passed through the chosen autoencoder (convolutional, LSTM, or linear). The autoencoder reconstructs the signal, converting the noisy 49 samples into denoised 49 samples, $\hat{x}(t)$ by leveraging the features it learned during training to mitigate AWGN channel noise. The denoised segments are then passed to the trained deep learning classifiers, which map these segments to their corresponding binary value, $\hat{b}_{(}i)$ (0 or 1) . These classifiers, which may be LSTM-based or Convolutional Neural Networks, are designed to decode the temporal or spatial patterns present in the denoised signal.

The classification process applies a thresholding mechanism to the predicted probabilities for each bit:

$$\hat{b}_i = \begin{cases} 1, & \text{if } P(r = 1 \mid X; \theta) \geq 0, \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

where,

- $P(r = 1 \mid X; \theta)$: The probability of the bit being 1, computed by the classifier.
- X: Denoised 49-sample feature vector.
- θ: Learned parameters of the classifier

This pipeline ensures that the signal undergoes significant noise reduction before classification, leading to enhanced accuracy in bit prediction. By mapping the denoised segments to binary values, the system achieves robust decoding even in high-noise environments, significantly improving the overall bit error rate compared to traditional approaches.

3. RESULTS AND DISCUSSION

An open-source, data-intensive HPCC computing system with dual Intel Xeon E5-2640 processors and improved computation capability from NVIDIA Tesla K20 M GPUs was used for the data generation and training phase. In the experiments, neural networks were trained using 80% of the dataset, with the remaining 20% being used for validation [10]. The trained autoencoder models were saved in .pth format, commonly used in PyTorch for preserving model states and weights. Similarly to this, the deep learning classifiers were stored in the HDF5 format utilizing a hierarchical data structure, which allowed for the safe and effective handling and storage of test data using Python's NumPy module. Most importantly, the autoencoders were trained on the same training dataset that was previously used for deep learning classifiers, but the testing was done on a completely different dataset. The results show that the system's BER performance has been significantly improved with the addition of autoencoders.

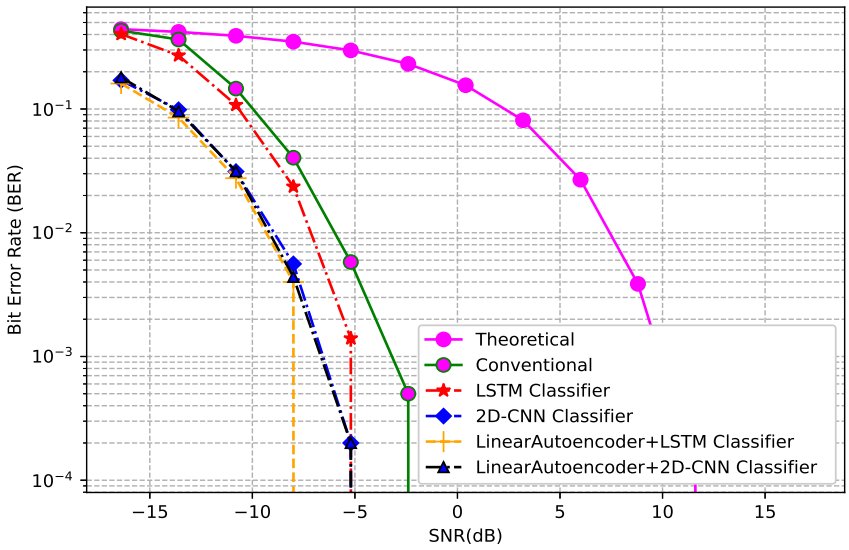

Figure 3: BER vs SNR comparison for receiver configurations with and without Linear Autoencoders

In this section, we present the performance evaluation of the proposed deep learning-based receiver models compared to the conventional correlation-based receiver and the theoretical BER performance of BPSK over an AWGN channel. The results are analyzed in terms of the Bit Error Rate (BER) versus Signal-to-Noise Ratio (SNR) for different models, as shown in FIGURES 3, 4, and 5. The theoretical BER performance of BPSK serves as a benchmark for assessing the

efficiency of the designed deep learning-based receivers. The conventional receiver closely follows the theoretical curve at moderate-to-high SNR levels but exhibits a degradation in performance at lower SNRs. This observation aligns with the expected behavior of correlation-based detection under noisy conditions.

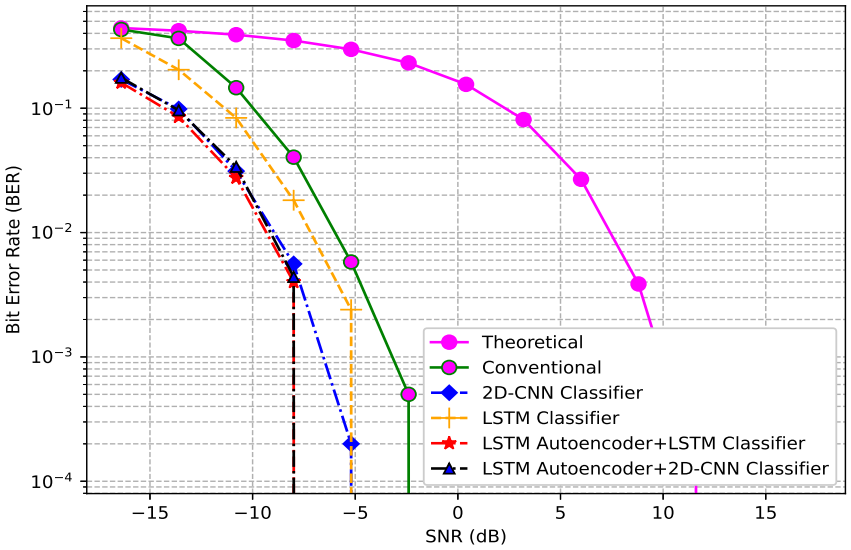

Figure 4: BER vs SNR comparison for receiver configurations with and without LSTM Autoencoders

The proposed machine learning models, including the LSTM classifier, the 2D-CNN classifier and the architectures based on auto-encoders, demonstrate significant improvements in BER performance over the conventional receiver. The 2D-CNN classifier shows notable robustness, achieving a better BER at lower SNRs compared to the LSTM classifier. Among the models based on autoencoders, the hybrid approaches integrating convolutional and recurrent architectures outperform their standalone counterparts, indicating the benefit of feature extraction combined with sequence learning. The FIGURE 3, compares the results obtained with and without the integration of linear autoencoders, demonstrating the performance of the Bit Error Rate (BER) of a BPSK communication system across a range of SNR. The standalone deep learning classifiers, such as 2D-CNN and LSTM, show larger BER values in the absence of autoencoders, especially in low-SNR regions where noise predominates. However, BER is greatly decreased across all SNR ranges by integrating autoencoders. In situations with moderate to high SNR, the Linear Autoencoder efficiently lowers BER despite being computationally simpler. These findings demonstrate how well linear autoencoders mitigate noise when used in conjunction with deep learning classifiers, bringing BER performance closer to the theoretical standard.

The plot FIGURE 4, evaluates the receiver configurations with and without the inclusion of the LSTM Autoencoder. Compared to the performance of a Linear Autoencoder, the results highlight the superior capabilities of the LSTM autoencoder, particularly in mitigating sequential noise and improving decoding accuracy across a wide range of SNR conditions. Both classifiers alone struggle

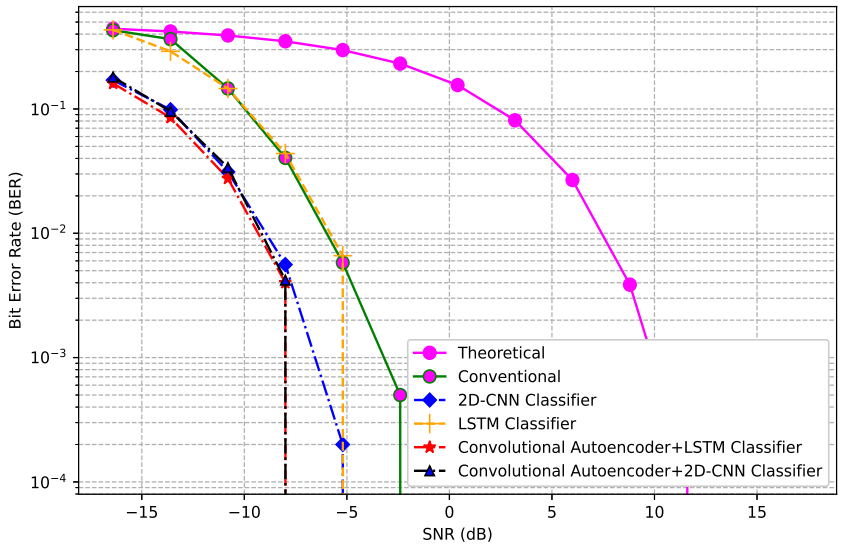

Figure 5: BER vs SNR comparison for receiver configurations with and without Convolutional Autoencoders

in low-SNR regions due to the lack of effective denoising mechanisms, resulting in higher BER values.

When the LSTM Autoencoder is integrated into the pipeline , the BER is significantly reduced across all SNR levels. This improvement is most evident in low- and mid-SNR regions, where the LSTM Autoencoder excels in leveraging its sequential learning capabilities to model and mitigate temporal noise patterns. By reconstructing cleaner signal representations before passing them to the classifiers, the LSTM Autoencoder enables the system to achieve BER performance closer to the theoretical limit.

The FIGURE 5, illustrates the Bit Error Rate (BER) performance of a BPSK communication system across various Signal-to-Noise Ratios (SNR), with and without the integration of the 2D-Convolutional Autoencoder. With the inclusion of the 2D-Convolutional Autoencoder, there is a substantial reduction in BER across all SNR ranges. The most notable improvements occur in high-SNR regions, where the 2D-Convolutional Autoencoder effectively extracts spatial features from the received signals, enabling better noise suppression. Its ability to handle structured or spatially distributed noise makes it particularly effective in scenarios where the received signals have complex spatial patterns.

The enhanced learning and generalization capabilities of neural networks in differentiating between noise and signal are the reason for the abrupt drop in BER for deep learning-based models at high SNR. The model more successfully learns the underlying signal structure and reduces noise as SNR rises. The decision border becomes well defined at high SNRs, enabling nearly flawless classification with few bit mistakes. Because deep learning models employ sigmoid-based or softmax

decision thresholds, the probability of correctly classifying a bit increases significantly, leading to an abrupt drop in BER as misclassifications.

4. CONCLUSIONS

This work offers a transformative advancement in communication receivers, particularly in challenging noise environments, by integrating autoencoders with deep learning classifiers. We have evaluated the distinct advantages of linear, LSTM, and convolutional autoencoders and as well as their capacity for signal denoising. The findings demonstrate that all models significantly improve Bit Error Rate performance across a broad range of Signal-to-Noise Ratios. These findings demonstrate the substantial improvements achieved over conventional receivers, particularly under additive white Gaussian noise. This demonstrates how adaptably deep learning and adaptive preprocessing layers can be combined to create reliable, noise resilient communication systems.

The practical viability of this methodology further enhances its appeal. The availability of open-source tools for real-time signal processing and development kits for software-defined radios (SDRs) makes a physical implementation of this system possible. This makes the suggested layout a useful, approachable, and scalable solution for contemporary communication systems in addition to being a theoretical improvement. Looking ahead, future research could explore the incorporation of sophisticated designs like Transformer-based models or attention techniques. Multi-modal approaches that incorporate diverse signal characteristics–such as phase, amplitude, and frequency–present exciting opportunities for adaptation in complex and dynamic channel environments. Additionally, optimizing the system for deployment on resource-constrained devices while enabling real-time adaptability could pave the way for intelligent, energy-efficient receivers. These future explorations promise to redefine the boundaries of mission-critical, adaptive, and efficient communication technologies.adaptive, and efficient communication technologies.

References

[1] Sabella D, Micheli D, Nardini G. The Power of Data: How Traffic Demand and Data Analytics Are Driving Network Evolution Toward 6G Systems. J Sens Actuator Netw. 2023;12:49.

[2] Zuo X, Yang Y, Yao R, Fan Y, Li L. An Automatic Modulation Recognition Algorithm Based on Time Frequency Features and Deep Learning With Fading Channels. Remote Sens. 2024;16:4550.

[3] Farsad N, Goldsmith A. Neural Network Detection of Data Sequences in Communication Systems. IEEE Trans Signal Process. 2018;66:5663-5678.

[4] Wang B, Xu K, Zheng S, Zhou H, Liu Y. A Deep Learning-Based Intelligent Receiver for Improving the Reliability of the Mimo Wireless Communication System. IEEE Trans Reliab. 2022;71:1104-1115.

[5] Wu T. CNN and RNN-Based Deep Learning Methods for Digital Signal Demodulation. In: Proceedings of the 2019 international conference on image video and signal processing. New York USA: ACM. 2019:122-127.

[6] Kalinin VI, Byshevski-Konopko OA, Kotelnikov IR. Fluctuations of Correlation Performances in Digital Noise Communications. J Radio Electronics. 2024.

[7] Buduma N, Buduma N, Papa J. Fundamentals of Deep Learning. O'Reilly Media Incorporated; 2022.

[8] Chollet F, Chollet F. Deep Learning With Python. Simon & Schuster. 2021.

[9] Honkala M, Korpi D, Huttunen JM. Deeprx: Fully Convolutional Deep Learning Receiver. IEEE Trans Wirel Commun. 2021;20:3925-3940.

[10] Zheng S, Chen S, Deepreceiver YX. A Deep Learning-Based Intelligent Receiver for Wireless Communications in the Physical Layer. IEEE Trans Cogn Commun Netw. 2020;7:5-20.

[11] Wang B, Xu K, Song P, Zhang Y, Liu Y, et al. A Deep Learning-Based Intelligent Receiver for OFDM. In: 18th International Conference on Mobile Ad Hoc and Smart Systems. IEEE. 2021:562-563.

[12] Ying S, Huang S, Chang S, Yang Z, Feng Z, et al. A Convolutional and Transformer Based Deep Neural Network for Automatic Modulation Classification. China Commun. 2023;20:135-147.

[13] Vijay EV, Aparna K. Deep Learning-Ct Based Spectrum Sensing for Cognitive Radio for Proficient Data Transmission in Wireless Sensor Networks. e-Prime-Advances in Electrical Engineering Electronics and Energy. 2024;9:100659.

[14] Qin Z, Ye H, Li GY, Juang BH. Deep Learning in Physical Layer Communications. IEEE Wirel Commun. 2019;26:93-99.

[15] O'shea T, Hoydis J. An Introduction to Deep Learning for the Physical Layer. IEEE Trans Cogn Commun Netw. 2017;3:563-575.

[16] Ye H, Li GY, Juang BH. Power of Deep Learning for Channel Estimation and Signal Detection in Ofdm Systems. IEEE Wirel Commun Lett. 2018;7:114-117.

[17] Samuel N, Diskin T, Wiesel A. Deep Mimo Detection. In: IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). 2017:1-5.

[18] Huang H, Guo S, Gui G, Yang Z, Zhang J, et al. Deep Learning for Physical-Layer 5G Wireless Techniques: Opportunities, Challenges and Solutions. IEEE Wirel Commun. 2019;27: 214-222.

[19] Varsha PS, Hari VS. Intelligent Deep Learning Based Speech Receivers. IRECAP. 2023;13:189-196.

[20] Subray S, Tschimben S, Gifford K. Towards Enhancing Spectrum Sensing: Signal Classification Using Autoencoders. IEEE Access. 2021;9:82288-82299.

[21] Chen S, Guo W. Auto-Encoders in Deep Learning a Review With New Perspectives. Mathematics. 2023;11:1777.

[22] Rao DJ, Prakasam V. Design and Implementation of Ds-CDMA Modem Using BPSK. Int J Sci Eng Technol Res. 2017;6:1432-1439.

[23] Cubukcu E. Root Raised Cosine (RRC) Filters and Pulse Shaping in Communication Systems. In: AIAA Conference JSC-CN. 2012;26387.

[24] Varsha PS, Hari VS. LSTM Based Receiver Design for Baseband Signal Demodulation. In: International Conference on Computing, Communication, and Intelligent Systems (ICCCIS). 2022:713-718.

[25] Hernández-Vázquez A, Hernández-Rodríguez Y, Cortes-Rojas F, Bayareh-Mancilla R, Cigarroa-Mayorga O. Automated Mammogram Analysis for Microcalcification Identification Using Convolutional Neural Networks. In: Congreso nacional de ingeniería biomédica. Springer. 2024:113-123.

[26] Michelucci U. An Introduction to Autoencoders. 2022. Arxiv preprint https://arxiv.org/pdf/2201.03898

[27] Berahmand K, Daneshfar F, Salehi ES, Li Y, Xu Y. Autoencoders and Their Applications in Machine Learning: A Survey. Artif Intell Rev. 2024;57:28.

[28] Li P, Pei Y, Li J. A Comprehensive Survey on Design and Application of Autoencoder in Deep Learning. Appl Soft Comput. 2023;138:110176.

[29] Park J, Seok J, Hong J. Autoencoder-Based Signal Modulation and Demodulation Methods for Sonobuoy Signal Transmission and Reception. Sensors. 2022;22:6510.

[30] Xue J, Huang Q, Wu S, Nagao T, Lstm-Autoencoder Network for the Detection of Seismic Electric Signals. IEEE Trans Geosci Remote Sens. 2022;60:1-12.

[31] Sah NK, Kolli M, Dharmaraj KP, Vishwas HN. Comparative Deep Learning Approach for Intrusion Detection. In: 15th International Conference on Computing Communication and Networking Technologies (ICCCNT). 2024:1-6.

[32] Alnaseri O, Alzubaidi L, Himeur Y, Timmermann J. A Review on Deep Learning Autoencoder in the Design of Next-Generation Communication Systems. 2024. Arxiv preprint: https://arxiv.org/pdf/2412.13843v1

[33] Acharjee R, Ahamed SR. Automatic Eyeblink Artifact Removal From Single Channel Eeg Signals Using One-Dimensional Convolutional Denoising Autoencoder. In: International Conference on Computer Electrical & Communication Engineering (ICCECE). IEEE. 2024:1-7.