Sparse Autoencoder-Based Deep Learning Approach for Enhancing End-to-End Communication System Performance

Safalata S Sindal
Dept. of Electronics & Communication Engineering
Institute of Technology, Nirma University
Ahmedabad, Gujarat, India
22ftphde68@nirmauni.ac.in

Y N Trivedi
Dept. of Electronics & Communication Engineering
Institute of Technology, Nirma University
Ahmedabad, Gujarat, India
yogesh.trivedi@nirmauni.ac.in

***Abstract*—Deep learning is increasingly adopted in future communication systems to meet requirements within constrained resources. End-to-end (E2E) autoencoder models leverage deep neural networks for effective communication but often suffer from performance degradation due to overfitting. To overcome this challenge, we propose the use of a sparse autoencoder (SAE). The SAE captures essential features by enforcing sparsity, enhancing model generalization. Building on this approach, we propose an SAE-based E2E communication system model designed for M-ary Phase Shift Keying (MPSK) and M-ary Quadrature Amplitude Modulation (MQAM) constellations over an Additive White Gaussian Noise (AWGN) channel. We train the SAE under varying signal-to-noise ratios (SNRs) and evaluate its Block Error Rate (BLER) versus testing SNR performance. Results show improved BLER at lower training SNRs, as the model better learns noise behavior. The proposed system outperforms a conventional BPSK system using maximum likelihood detection and a baseline E2E autoencoder, demonstrating the effectiveness of sparsity constraints. Additionally, we assess its performance considering model loss function, further validating its robustness.**

Keywords—**deep learning, sparse autoencoder, end-to-end communication, overfitting.**

I. Introduction

Deep Learning (DL) has emerged as a transformative approach in artificial intelligence, demonstrating remarkable capabilities in solving complex problems once considered unattainable. Its ability to automatically learn intricate patterns from large datasets has driven significant advancements in diverse fields, including computer vision, autonomous systems, healthcare, and communication systems. It holds the potential to revolutionize the way we design, optimize, and operate communication systems [1]. By leveraging neural networks (NNs) and data-driven approaches, DL techniques have shown remarkable promise in enhancing various aspects of communication systems, such as modulation, demodulation, channel decoding, signal processing, and many more [2].

Conventional communication systems, while well-established and robust, often rely on manually designed rules and strategies that may not fully exploit the complexities of real-world communication environments [3]. In contrast, DL offers a data-driven approach capable of learning intricate patterns, adapting to dynamic conditions, and enhancing system performance through data analysis. Beyond improving traditional communication blocks, DL introduces a new paradigm for future systems, where features and parameters are learned directly from data without manual or ad-hoc intervention, using an end-to-end (E2E) loss function.

Inspired by this methodology, E2E learning-based communication system was introduced in [4], where both the transmitter and receiver are represented by deep NNs, interpreted as autoencoders (AEs). The study demonstrated comparable performance to traditional communication systems; however, the input data used had a restricted bit length. This constraint was later addressed by [5] and [6]. The work in [5] leveraged a convolutional NN-based AE with enhanced generalization capabilities, allowing support for arbitrary input bit lengths. Despite this advancement, their loss curve exhibited oscillatory behavior, indicating potential instability in the training of an overly complex model. Conversely, [6] introduced a conditional generative adversarial network (GAN) to manage dimensionality issues in long transmit symbol sequences. While their model demonstrated stable training, the use of a GAN as a channel model introduced added complexity due to its deep architecture. To mitigate this complexity, [7] proposed a compression algorithm that pruned less significant weight coefficients during training. This approach effectively reduced model complexity; however, it led to performance degradation at high SNR levels. In [8], a residual AE with a convolutional block attention scheme was used at the decoder to extract fine-grained features for noise reduction. The models in [4]-[8] achieve comparable performance to the conventional communication systems, while [9] showed a performance gain over conventional systems using a fully differentiable neural iterative demapping and decoding AE structure. However, their achieved gain is minimal. All these models employ complex NN architecture using multiple layers.

As NNs grow in complexity and size, they become more prone to overfitting i.e. excelling at training data but struggling to generalize to unseen data, thus limiting their flexibility and

adaptability. To mitigate this, extensive research has explored regularization techniques to enhance robustness and generalization in NNs. In [10], overfitting analysis for transmission systems over an AWGN channel was demonstrated, showing improved performance with regularization. However, in scenarios with limited training data or noisy input, basic AEs with regularization may still struggle to extract relevant features, as regularization penalties on weights do not necessarily prioritize the most informative aspects of the data.

To combat overfitting, various regularization techniques have been explored [11]. Dropout, as applied in [12], enhances NN adaptability by probabilistically removing neurons, while data augmentation in [13] reduces overfitting by expanding the training set without adding new information. One such regularization technique gaining prominence is the sparse autoencoder (SAE) [14], known for capturing essential features and reducing data dimensionality [15]. SAEs enforce sparsity constraints on neuron activations, encouraging the network to learn more concise and informative representations of the input data. The key contributions of this research are outlined below to highlight the significance of the proposed approach:

- We propose a DL-based E2E communication system over an AWGN channel to addresses the challenge of overfitting by applying an SAE as a regularization technique.
- We design and present the algorithm to describe the functionality and training process of the SAE.
- We evaluate the system performance using Block Error Rate (BLER) versus average SNR (E_b/N_0) to assess reliable communication under different noise levels.
- We compare the performance of the proposed model with conventional M-PSK and M-QAM systems employing maximum likelihood detection (MLD), highlighting the advantages in various modulation scenarios.
- We benchmark the proposed model against an existing AE-based system to demonstrate its performance gains.
- We demonstrate the effectiveness of the proposed model in enhancing the performance of E2E communication systems with M-PSK and M-QAM modulations across different training SNR conditions.

The rest of the paper's organization is as follows: Section II presents the conventional AWGN communication model along with its equivalent DL-based E2E communication system model. In Section III, the simulation results of the proposed SAE-based system are presented using BLER. Finally, we conclude the paper in Section IV.

II. System Model

This section presents the AE-based communication system and explains the functionalities of a basic AE and SAE. The training process of SAE has also been explained in detail.

A. Autoencoder Based End-to-End Communication System

A conventional communication system with AWGN channel [3], shown in Fig. 1, can be modeled as $\mathbf{y} = \mathbf{s}+\mathbf{w}$, where $\mathbf{y}$, $\mathbf{s}$, and $\mathbf{w}$ represent the symbols of the received signal, transmitted signal, and additive white Gaussian noise, respectively. The

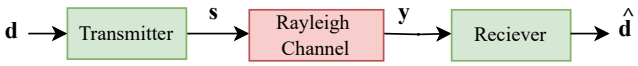

Fig. 1. A conventional communication system model

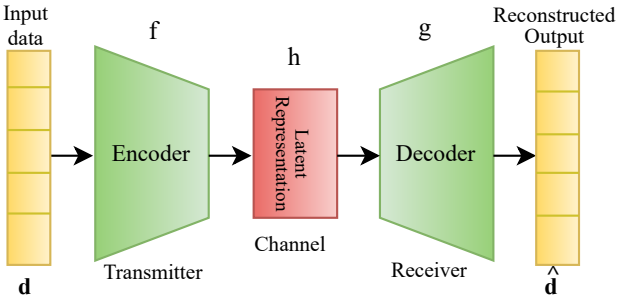

Fig. 2. Representation of an Autoencoder as a communication system model

energy of $\mathbf{s}$ is E_b and the power spectral density of $\mathbf{w}$ is N_o. The system coding rate is defined as $R = k/N$ [bit/channel use], where N is the number of channel uses, $k = \log_2(M)$, and M is the number of possible messages being transmitted for the MPSK system. The goal of the transmitter is to send a single message from a set of M possible messages, represented as $\mathbf{d} \in \mathcal{M} = \{1, 2, ..., M\}$, over an AWGN channel to the receiver using N discrete channel transmissions. To accomplish this, the transmitter encodes $\mathbf{d}$ into a transmitted signal $\mathbf{s} \in \mathbb{R}^N$. The channel behavior is characterized by the conditional probability density function $p(\mathbf{y}|\mathbf{s})$. At the receiver, the received noisy signal is processed to obtain the estimated message $\hat{\mathbf{d}}$. The performance of the system is assessed using the BLER, defined as $Pr(\hat{\mathbf{d}} \neq \mathbf{d})$.

We have modeled this conventional communication system as an SAE-based E2E communication system. The encoder corresponds to the transmitter, with its input being the data to be transmitted. The encoder component of the SAE compresses this input into a lower-dimensional representation, aiming to capture essential information while discarding less relevant details. To counter channel noise effects, a receiver is represented by a decoder. The decoder reconstructs the original data from the received signal. The SAE is trained to encode the input data efficiently for transmission and decode the received data to reconstruct the original input. The reconstruction is done by adjusting the parameters during training to minimize the difference between input and output data.

Fig. 2 shows the representation of an AE as a communication system model. Encoder (f), which acts as a transmitter, compresses the input data $\mathbf{d}$ into a latent space representation (channel) $\mathbf{h}$, represented by an encoding function $\mathbf{h}$=f($\mathbf{d}$) [14]. Decoder (g), acting as a receiver, reconstructs the input from the latent space representation, defined by the decoding function $\hat{\mathbf{d}} = g(\mathbf{h})$. Consequently, the overall AE operation

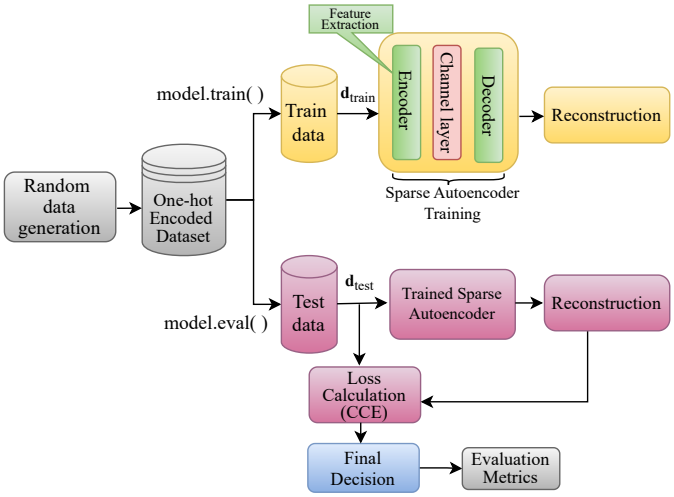

Fig. 3. Workflow of Proposed Sparse Autoencoder Model

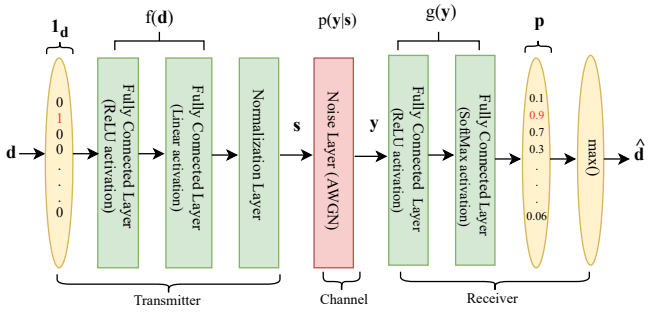

Fig. 4. Proposed Sparse Autoencoder based communication system model with AWGN channel

can be expressed as $\hat{\mathbf{d}} = g(f(\mathbf{d}))$, where the objective is to make $g(f(\mathbf{d}))$ as close as possible to the original input $\mathbf{d}$. Training a basic AE involves minimizing the reconstruction error, which quantifies how well the model can reconstruct the input data. The reconstruction error [14] for a basic AE is given by $L(\mathbf{d}, \hat{\mathbf{d}})$. AEs suffer from overfitting when the model learns to fit the training data too closely, capturing even the noise and minor variations present in the dataset. As a result, the model performs well on the training data but poorly on unseen data due to limited generalization. Regularization techniques are used in NNs to prevent overfitting [10]. We use an SAE [14] which is a type of regularized AE.

B. Sparse Autoencoder Training

Unlike a basic AE, an SAE learns only the most important features of its input. The SAE's loss function promotes sparsity and feature selectivity alongside reconstruction. Sparsity ensures that only a few neurons in a layer are activated, enhancing generalization, critical for communication system applications. By incorporating SAE, the model can capture key communication features such as modulation schemes, phase variations in MPSK models, and SNR.

We propose an SAE-based E2E communication system model, where the SAE introduces a sparsity constraint, i.e. a sparsity penalty term $\Omega(\mathbf{h})$ in the hidden layer $\mathbf{h}$. Thus, for the proposed SAE model, the reconstruction error becomes

$$L(\mathbf{d}, \hat{\mathbf{d}}) + \Omega(\mathbf{h}) \tag{1}$$

The sparsity penalty term promotes sparsity in $\mathbf{h}$. The reconstruction error quantifies the difference between the input $\mathbf{d}$ and the reconstructed output $\hat{\mathbf{d}}$. Eq. (1) shows how sparsity is encouraged in the latent representation $\mathbf{h}$ within the SAE framework. However, in our proposed SAE-based model, the regularization approach is designed to enforce sparsity specifically in the normalization layer, significantly influencing the latent representation structure. To incorporate the sparsity penalty into the loss function, we utilize L1 regularization [14]. This regularization technique promotes sparsity by penalizing nonzero activations, ensuring only a few neurons are active. The network minimizes the L1 regularization term along with the reconstruction error during training, helping mitigate overfitting and handle noisy datasets effectively. In the SAE framework, the forward pass, where input data propagates through network layers to generate a prediction or output, is expressed as

$$\mathbf{h} = f(W\mathbf{d} + b), \tag{2}$$

where $f(\cdot)$ represents the activation function, while W, $\mathbf{d}$ and b correspond to the weight matrix, input, and bias term, respectively. This computation is performed iteratively for each layer, starting from the input layer and proceeding through hidden layers until reaching the output layer.

The SAE workflow, illustrated in Fig. 3, consists of two main phases: training and testing. During the training phase, SAE learns from a dataset of encoded symbols. In the testing phase, the trained SAE is evaluated using a separate testing dataset to measure its performance. Fig. 4 shows the proposed SAE-based communication system architecture over an AWGN channel. The input data are generated as random binary data $\mathbf{d}$, encoded into a one-hot vector $\mathbf{1_d} \in \mathbb{R}^M$ (an M-dimensional vector), the $\mathbf{d}^{th}$ element of which is equal to one and zero otherwise [4]. This one-hot vector creates the actual input data for training. The transmitter maps $\mathbf{d}$ into N numbers using two fully connected (FC) layers with ReLU and linear activation functions [14], performing nonlinear and linear transformations for encoding and modulation. The normalization layer normalizes the output of the last FC layer to constrain the signal energy within the fixed value N. The $L1$ regularization is applied to neurons from the normalization layer for sparsity constraint. The channel is modeled as an additive noise layer with fixed variance $\beta = (N_0/2RE_b)$. The decoder corresponds to the receiver. The received signals are processed by two FC layers with ReLU and softmax activation [14]. The AE is trained iteratively using categorical cross-entropy loss and Adam optimizer [14]. The loss function measures the difference between the estimated message $\hat{\mathbf{d}}$

and the target input $\mathbf{d}$. A probability vector, representing the probabilities over all possible messages, is reconstructed to restore the original data. At last, the estimated message $\hat{\mathbf{d}}$ can be obtained by selecting the largest element index of the probability vector. The AE can then be trained end-to-end on the set of all possible messages $\mathbf{d} \in \mathcal{M}$ using the loss function.

Algorithm 1 SAE Training

Input:

- Training dataset $\mathbf{d}$
- SAE architecture: Encoder f and Decoder g.
- Hyperparameters: learning rate (α), number of epochs (N_epoch), number of training iterations per epoch (N_iter), $L1$ regularization strength (λ), and loss function L.

Initialization:

- Random initial AE parameters $\boldsymbol{\theta_f}$ and $\boldsymbol{\theta_g}$.
- Epoch counter: epoch = 0.

Training Loop:

for N_epoch **do**:

1: Shuffle $\mathbf{d}$ randomly.
2: $total\_loss = 0$.
3: $total\_l1\_reg = 0$.
4: **for** N_iter **do**:
   a: Batch selection: $\mathbf{d}\_batch$ from $\mathbf{d}$
   b: $z\_batch = f(\mathbf{d}\_batch; \boldsymbol{\theta_f})$
   c: $\hat{\mathbf{d}}\_batch = g(z\_batch; \boldsymbol{\theta_g})$
   d: $loss\_batch = L(\mathbf{d}\_batch, \hat{\mathbf{d}}\_batch)$
   e: $l1\_reg\_batch = \lambda \cdot \sum_i |(\boldsymbol{\theta_f})_i|$
   f: $total\_loss+ = loss\_batch$
   g: $total\_l1\_reg+ = l1\_reg\_batch$
   end for
5: $avg\_loss = total\_loss/N\_iter$
6: $avg\_l1\_reg = total\_l1\_reg/N\_iter$
7: Output: avg_loss and avg_l1_reg term for current epoch.
8: Total loss for current epoch: $total\_loss\_with\_l1 = avg\_loss + avg\_l1\_reg$
9: Gradients computation of total loss with respect to f and g parameters: $\nabla\boldsymbol{\theta_f} total\_loss\_with\_l1, \nabla\boldsymbol{\theta_f} avg\_loss$
10: Parameter update using gradient descent:
$\boldsymbol{\theta_f} = \boldsymbol{\theta_f} - \alpha \cdot (\nabla\boldsymbol{\theta_f} total\_loss\_with\_l1 + \lambda \cdot \text{sign}(\boldsymbol{\theta_f}))$
$\boldsymbol{\theta_g} = \boldsymbol{\theta_g} - \alpha \cdot \nabla\boldsymbol{\theta_g} avg\_loss$

end for

Output: Trained SAE parameters: $\boldsymbol{\theta_f}$ and $\boldsymbol{\theta_g}$.

Algorithm 1 outlines the training process of the SAE. The term N_iter is depends on the batch size and training data, while $\boldsymbol{\theta}$ denotes the trainable parameters (W and b) of a fully connected layer. To enforce sparsity, $L1$ regularization is applied to each batch size by scaling the encoder parameters $\boldsymbol{\theta_f}$ with λ, reducing the magnitude of larger parameter values. In step 4.e, i represents the index iterating over the individual elements in $\boldsymbol{\theta}$. By iteratively refining the SAE parameters $\boldsymbol{\theta_f}$ and $\boldsymbol{\theta_g}$, the model seeks to minimize the overall loss function and improve data reconstruction. In SAEs, inducing sparsity in the learned representations is a key objective. Step 10 reflects this objective by updating the encoder $\boldsymbol{\theta_f}$ parameters with an added term that encourages sparsity. The term $\lambda \cdot \text{sign}(\theta_f)$ acts as a sparsity-inducing penalty. This encourages a large portion of the encoder parameters to become zero, yielding a sparse representation. Once the SAE is trained, the trained model can be evaluated on unseen testing data to assess its performance. The simulation results are presented in Section III.

III. System Performance

A. Simulation Setup

This section presents simulations of the proposed SAE-based model. BLER is used as the performance metric during evaluation and testing. The training dataset consists of 16,000 independently and identically distributed integer samples, with separate datasets for training and testing to ensure distinct noise realizations. A 20% split of the training dataset is reserved for validation. Table I outlines the proposed SAE model structure. The output dimensions indicate the number of neurons in each layer. The modulation learning process begins in the third layer, where the output dimension is reduced to $N = 1$ for effective feature learning. After the noise layer, the output dimension is restored to M at the decoder. The values of M and N are chosen based on different M-PSK and M-QAM systems, with the coding rate R set to 1.

TABLE I
AUTOENCODER MODEL LAYOUT

Layer	Output Dimensions
Input	M
FC + ReLU	M
FC + Linear	N
Normalization	N
Noise	N
FC + ReLU	M
FC + Softmax	M

TABLE II
HYPERPARAMETERS USED FOR TRAINING

Hyperparameter	Value
Learning rate	0.001
Number of epochs	100
L1 regularization strength	0.01
Batch size	64
Optimizer	Adam
Loss function	Categorical Crossentropy

Table II provides all hyperparameters used in training the proposed model. All the values were chosen to ensure effective training convergence and performance of the proposed model. The $L1$ regularization strength (λ) was set to 0.01 to maintain a balanced trade-off between sparsity and reconstruction quality. Lower values (e.g., 0.005), reduced sparsity and increased the risk of overfitting, while higher values (e.g., 0.02) imposed excessive sparsity, leading to degraded performance.

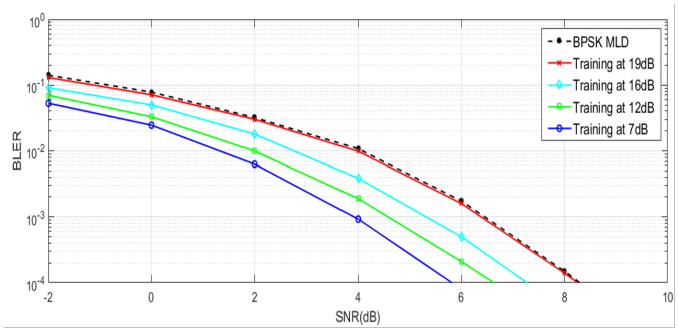

Fig. 5. BLER versus testing SNR performance of BPSK SAE-based system with AWGN channel at different training SNRs.

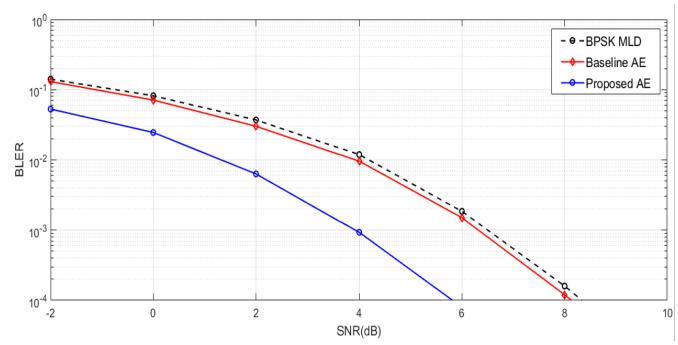

Fig. 6. BLER vs. testing SNR over AWGN channel for BPSK.

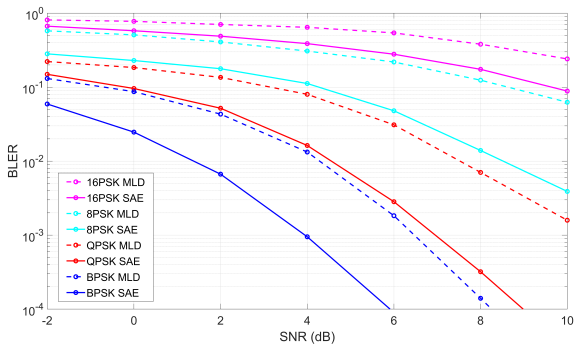

Fig. 7. BLER vs. testing SNR for the proposed SAE and the conventional MLD model for MPSK system, trained at an SNR of 7 dB.

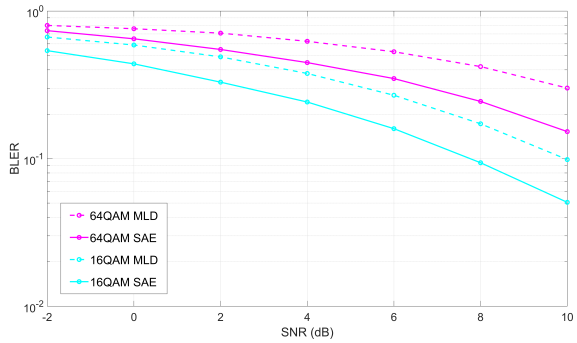

Fig. 8. BLER vs. testing SNR for the proposed SAE and the conventional MLD model for MQAM system, trained at an SNR of 7 dB.

B. Result Analysis

In Fig. 5, BLER versus testing average SNR of the proposed system for BPSK ($M = 2$) is shown for training SNRs of 7, 12, 16, and 19 dB. The proposed system performs better at lower training SNRs, as training at lower SNR levels enhances generalization and robustness against overfitting, improving performance on unseen data. However, this trend is evident only within the 7 to 19 dB range, with minimal change beyond this interval. We also compare the BLER performance conventional BPSK system in an AWGN channel using MLD, shown as BPSK MLD. The proposed system trained at 19 dB matches conventional BPSK, while lower training SNRs yield better performance. At 10^{-3} BLER, training at 7 dB provides a 2.5 dB SNR gain over conventional BPSK.

In Fig. 6, BLER versus testing SNR in AWGN is shown for the proposed system trained at 4 dB, compared with the conventional BPSK system. As discussed earlier, the proposed system outperforms the conventional one. We also include the BLER performance of the Baseline AE model [9], which lacks overfitting mitigation and performs optimally at 4 dB training SNR. The Baseline AE closely follows conventional BPSK across a wide SNR range, while the proposed system consistently outperforms the Baseline AE for all M values. At 10^{-3} BLER, the proposed model achieves approximately a 2 dB SNR gain over the Baseline AE, demonstrating the ability of SAE to capture essential features and mitigate overfitting.

Fig. 7 compares BLER versus testing SNR for the proposed SAE and conventional MLD models for $M = 2, 4, 8, 16$ in MPSK. In the conventional system, BLER performance degrades in AWGN for MPSK as M increases. A similar trend is observed in the proposed SAE system, which outperforms the conventional model for all M values.

TABLE III
GAIN OVER MLD SYSTEM AT 10^{-3} BLER

Values of M	Baseline AE	Proposed AE
2	0.3 dB	2.2 dB
4	0.4 dB	2.3 dB
6	0.7 dB	2.5 dB
8	0.8 dB	2.5 dB

Table III shows the SNR gain at 10^{-3} BLER, where the Baseline AE achieves a maximum gain of 0.8 dB for 8PSK, while the proposed model achieves up to 2.5 dB. Fig. 8 presents a similar comparison for MQAM, confirming the proposed SAE consistently outperforms the conventional system for all M values. Next, we analyze BLER versus training SNR at fixed testing SNR.

Fig. 9 illustrates this for BPSK at testing SNRs of 2, 4, 6, 8 dB. As expected, for a given testing SNR (e.g., 6 dB), BLER degrades with increasing training SNR, with a sharper decline at higher training SNRs, such as 8 dB. Fig. 10 shows the model loss curve for the proposed SAE system, illustrating

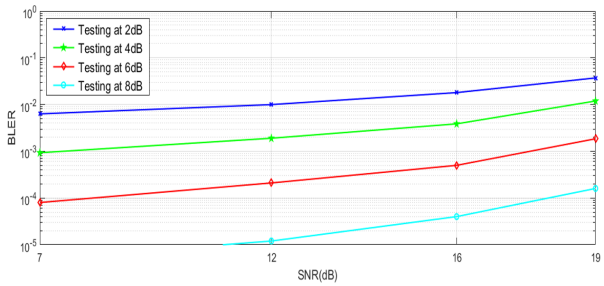

Fig. 9. BLER vs. training SNR for SAE-based BPSK at various testing SNRs.

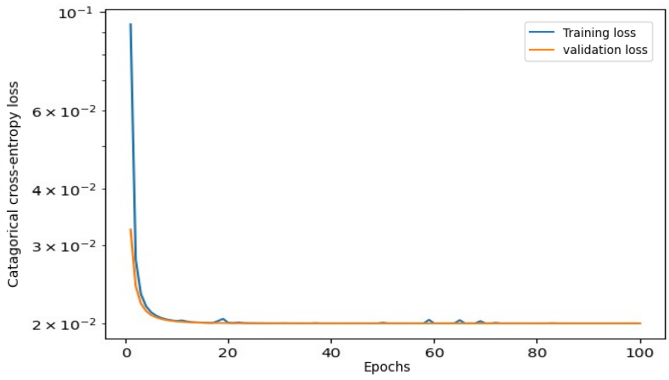

Fig. 10. Loss Curve of the Proposed SAE-Based System

its performance throughout training. Training loss reflects the model's ability to fit the training data, while validation (test) loss measures its generalization to unseen data. The curve, plotted over 100 epochs, shows a smooth and steady decline in training and validation loss, indicating effective learning and a stable training process, leading to improved performance of the proposed SAE system.

IV. Conclusion

End-to-end AE models often suffer from overfitting due to their complex structures. The proposed SAE-based model mitigates this issue by enhancing generalization, and improving performance. The model is trained using different SNR levels, achieving the best results at 7 dB training SNR. For BPSK, it offers a gain of about 2.5 dB over the conventional MLD approach. A comprehensive evaluation of MPSK and MQAM modulation schemes confirms the effectiveness of SAE in feature extraction and overfitting prevention. The proposed work advances efforts toward more efficient E2E communication systems. Future research can extend the SAE-based model to MIMO systems by adapting the encoder–decoder structure to support multiple antennas and integrating spatial encoding layers. The channel model can be expanded to include MIMO fading matrices, while training strategies can incorporate various antenna correlation and channel conditions. Advanced channels, like frequency-selective or time-varying fading, can be addressed by integrating memory-based modules such as attention mechanisms to improve robustness.

References

[1] E. Nachmani, Y. Be'ery and D. Burshtein, " Learning to decode linear codes using deep learning," IEEE Annual Allerton Conf. on Comm., Control, and Computing, pp. 341-346, 2016.

[2] I. Žeger and G. Šišul, "Introduction to Deep Learning Possibilities in Communication Systems," IEEE International Symp. ELMAR, pp. 21-24, 2021.

[3] J. Proakis and M. Salehi, "Fundamentals of Communication Systems," Pearson Education,2007.

[4] T. O'shea and J. Hoydis, "An introduction to deep learning for the physical layer," IEEE Trans. on Cognitive Comm. and Net., vol. 3(4), pp. 563-575, 2017.

[5] N. Wu Nan, X. Wang, B. Lin and K. Zhang, "A CNN-Based End-to-End Learning Framework Toward Intelligent Communication Systems," IEEE Access, vol. 7, pp. 110197-110204, 2019.

[6] H. Ye, L. Liang, G. Li and B. Juang, "Deep Learning-Based End-to-End Wireless Communication Systems With Conditional GANs as Unknown Channels," IEEE Tran. on Wireless Comm., vol. 19(5), pp. 3133-43, 2020.

[7] Y. Liang, C. Lam, and B. Ng, "Compression Algorithm for End-to-End Communication using CNN," 7th Inter. Conf. on Computer and Comm., pp. 318-323, 2021.

[8] M. Lu M, B. Zhou, and Z. Bu, "Attention-Empowered Residual Autoencoder for End-to-End Communication Systems," IEEE Comm. Letters, vol. 27(4), pp. 1140-1144, 2023.

[9] S. Cammerer, F. Aoudia, S. Dörner, M. Stark, J. Hoydis, and S. Brink, "Trainable Communication Systems: Concepts and Prototype," IEEE Trans. on Comm., vol. 68(9), pp. 5489-5503, 2020.

[10] H. Zhang, L. Zhang, and Y. Jiang, "Overfitting and underfitting analysis for deep learning based end-to-end communication systems," IEEE, 11th International Conf. on Wireless Comm. and Sig. Proc., pp. 1-6, 2019.

[11] H. Li, J. Li, X. Guan, B. Liang, Y. Lai, and X. Luo, "Research on overfitting of deep learning," IEEE 15th international conf. on computational intelligence and security, pp. 78-81, 2019

[12] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," Journal of Machine Learning Research, vol. 15(56), pp. 1929–1958, 2014.

[13] L. Perez L and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," Conv. Neural Net. Vision Recog., vol. 11, pp. 1-8, 2017.

[14] I. Goodfellow Ian, Y. Bengio Yoshua, and A. Courville, "Deep learning," MIT press, 2016.

[15] A. Ali and F. Yangyu, "Automatic modulation classification using deep learning based on sparse autoencoders with nonnegativity constraints," IEEE Sig. Proc. letters, vol. 24(11), pp. 1626-1630, 2017.