Performance Analysis of Autoencoders in Wireless Communication Systems with Deep Learning Techniques

K.Srinivasa Rao[1], R.K. Goswami[2], S.V.Rama Rao[3],Koteswararao Seelam[4]

[1]Professor, ECE, Dhanekula Institute of Engineering and Technology, Vijayawada, A.P.

[2]Professor, ECE, Gayatri Vidya Parishad College of Engineering for Women, Visakhapatnam, A.P.

[3]Associate Professor, ECE, NRIIT, Vijayawada, A.P.

[4]Professor, ECE, Kallam Haranath Reddy Institute of Technology, Guntur, A.P.

Abstract: Wireless experts worldwide have become interested in using Autoencoders (AEs) for modelling communication systems as an end-to-end reconstruction task. This approach optimizes both the transmitter and receiver components simultaneously, offering flexibility and convenience for representing complex channel models. Traditional communication systems rely on conventional models and assumptions that limit their utilization of limited frequency resources and hinder their ability to adapt to new wireless applications. However, with the rise of Artificial Intelligence, new wireless systems are capable of learning from wireless spectrum data and optimizing their performance. In this paper, the use of deep learning with autoencoders is explored to create an end-to-end communication system that replaces traditional transmitter and receiver activities. The autoencoder architecture effectively addresses channel impairments and enhances overall performance. Simulation results indicate that autoencoders surpass conventional communication systems in terms of Block Error Rate performance, even when facing impairments in the autoencoder's channel layer and using different neural network optimization algorithms.

Index Terms—Deep learning, autoencoders, wireless systems, physical layer, channel estimation.

1. Introduction

The transformational impact of wireless communication and related services on modern digital society cannot be overstated. However, emerging technologies such as smart cities, autonomous vehicles, and remote medical diagnosis pose challenges to traditional communication methods in terms of reliability, flexibility, energy efficiency, latency, and connection density. To meet these challenges, novel architectures, approaches, and algorithms are necessary at all layers of the communication system. In the past decade, machine learning, particularly deep learning, has been widely applied in various fields, including wireless communication [1, 2]. Researchers have investigated the applications of ML algorithms in channel coding, decoding, MIMO detection, and communication systems [3-9]. The communication field has a wealth of expert knowledge in information theory, probability, statistics, and mathematical modelling, with many approaches demonstrated for the physical layer, channel modelling [10], and optimal signalling [11]. The main purpose of a communication system is to send a message, like a stream of bits, accurately from the source to the destination through a channel, using a transmitter and receiver. For optimal performance, the transmitter

and receiver are divided into multiple independent blocks, each responsible for a specific task like channel coding, modulation, demodulation, or channel estimation [11]. While this block-based approach allows for individual optimization and control of each block, it may not always lead to optimal performance. According to [12], the block-based approach is sub-optimal in certain cases. However, a communication system based on deep learning optimizes the transmitter and receiver, without the need for separate blocks, following the traditional design of the communication system [12][13].

In this paper a novel approach to communication systems that utilizes deep learning has been introduced. Instead of employing separate encoding and decoding modules, this approach utilizes an autoencoder, which is a deep neural network comprising of an encoder and a decoder. The encoder learns a latent representation of the data, which is subsequently used by the decoder to reconstruct the input data. The authors suggest using an autoencoder to jointly optimize the communication between the transmitter and receiver, rather than optimizing their individual modules. The proposed design uses a convolutional encoder-decoder that considers channel impairments and optimizes the transmitter and receiver operations jointly for a single-antenna system. We evaluated how well our end-to-end AE performs in terms of block error rate (BLER) on an additive white Gaussian noise (AWGN) channel. The simulation results indicate that the AE-based model proposed has a Block Error Rate (BLER) that is similar to the conventional models that use modulation methods like BPSK and 16PSK. Furthermore, the study demonstrates that the proposed model has a better BLER compared to previous studies (references 12, 14, and 15). These findings demonstrate the possibility of using AE-based end-to-end communication systems as a substitute for standard block-based wireless communication systems.

The paper is structured as follows: Section 2 examines relevant literature. Section 3 offers a concise introduction to the AE-based communication system and examines regularization. Section 4 outlines the proposed model, while Section 5 details the simulations and performance evaluation of the implemented AE system. Lastly, Section 6 concludes the paper.

2. Related works

T. O'Shea et al. introduced the idea of employing autoencoders (AEs) in communication systems, as stated in their works [12] and [13]. In [12], the authors view the communication system as an AE and propose an approach to design a communication system as an end-to-end reconstruction task, which involves optimizing both the transmitter and receiver components simultaneously in a single process. They utilize a feedforward neural network to replace the functions of the transmitter and receiver. In [13], the primary approach for developing end-to-end radio communication systems is through the utilization of the AE channel. The authors tackle the task of learning as an unsupervised machine learning problem and concentrate on enhancing the reconstruction loss by introducing synthetic impairment layers. They include various regularizing layers that simulate the typical impairments encountered in wireless channels. Additionally, [17] examines an optical wireless

communication system that serves a single user, utilizing AEs. In conditions where the channel response is unknown or not easily modelled, the authors in [19] proposed an extended channel AE model for end-to-end learning. An adversarial approach was used to approximate the channel response and encode information, allowing both tasks to be learned simultaneously over a wide range of channel conditions. The authors demonstrated the effectiveness of this model in an over-the-air system through training and validation. Another study in [20] investigated the impact of optimizers on AE convergence speed for high-mobility and short-coherence channel applications. End-to-end learning has also been applied in molecular and optical communications with promising performance, indicating the potential of deep learning in complex communication scenarios [21, 22]. Furthermore, we assess various channel uses and modulation techniques in our design and the constellations produced by the autoencoder. The findings highlight the effectiveness of optimizing with deep learning techniques in creating innovative methods for wireless communication design.

3. Channel autoencoder in wireless communications

3.1. Conventional wireless Communication system

A wireless communication system typically includes three components: a transmitter, channel, and receiver. The transmitter sends a message *s*, selected from a set of *M* possible messages $s \in M = \{1, ..., M\}$, to the receiver through *n* uses of the channel. The message *s* is subjected to digital modulation $f: M \mapsto R_n$, resulting in a vector $x = f(s) \in R^n$ that is transmitted. This modulation maps the input symbols from a discrete alphabet to complex numbers that indicate points on the constellation diagram. The transmitter imposes power constraints on *x*, such as an energy constraint $\|x\|_2^2 \leq n$ or an average power constraint $E[|x_i|^2] \leq 1$ for all *i*. Each message *s* can be represented using $k = log_2(M)$ bits, so the system operates at a communication rate of $R = k/n$, measured in bits/channel use. The channel introduces distortions to the transmitted symbols. Upon reception, the receiver produces an estimate $\hat{s}$ of the originally transmitted message *s*. The Block Error Rate (BLER) *Pe* can be defined as the probability that $\hat{s}$ does not match *s* as given below (1).

$$P_e = \frac{1}{M}\sum_s P_r(\hat{s} \neq s/s) \quad (1)$$

As shown in Fig. 1, the conventional communication system consists of multiple independent blocks. The source encoder compresses the input data and eliminates redundancy, while the channel encoder adds controlled redundancy to the output of the source encoder. Channel coding, also known as forward error correction, is typically used in wireless communication systems to ensure that the received data is the same as the transmitted data, as wireless links are prone to fading and interference, which can lead to errors. To overcome this, the transmitter adds extra information to the data before sending it, a process called coding. This helps to mitigate the adverse effects of the communication medium. An uncoded communication system, on the other hand, does not include additional information to mask the data being sent. The modulator block changes the characteristics of the signal based on the selected data rate and the signal level received at the receiver,

provided that the modulation method used at the transmitter is adaptable. The channel distorts and weakens the transmitted signal before additional noise is added due to hardware impairments when the signal reaches the receiver. Each communication block at the transmitter prepares the signal to withstand the effects of the communication medium and receiver noise while maximizing system efficiency. The receiver performs similar operations in reverse order to reconstruct the transmitted information.

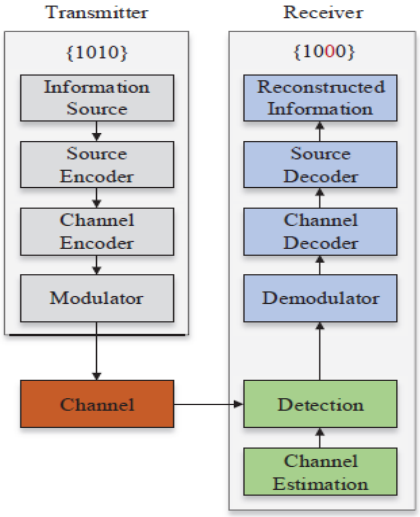

Fig. 1. A conventional wireless communication system model illustrating channel coding and modulation blocks.

3.2. An End-to-end Optimization Process with Autoencoder

An autoencoder (AE) is a type of Feed-forward Neural Network (FNN) where the input and output are equivalent. The original AE is an unsupervised deep learning algorithm that compresses the inputs to learn a reduced representation, which can be used to reconstruct the original inputs at the output layer [23]. The AE has a hidden layer that represents the input code. The network typically consists of two parts: an encoder function $y = f(s)$ that converts the input s into a compressed form y, and a decoder function $r = g(y)$ that produces a reconstruction r from y. The simplest form of an AE consists of one hidden layer and is defined by two weight matrices W and two bias vectors b.

$$y = f(x) = s_1\left(W^{(1)}x + b^{(1)}\right), \tag{2}$$

$$r = g(y) = s_2\left(W^{(2)}y + b^{(2)}\right), \tag{3}$$

where s_1 and s_2 represents the activation functions, which are generally nonlinear.

Thus, it is possible to view the communication system as an Autoencoder (AE) that aims to reconstruct the transmitted messages at the receiver with minimal error. The encoder and decoder can be seen as performing the functions of the transmitter and receiver, respectively. A typical AE structure that can be utilized for end-to-end learning of a communication system is illustrated in Fig. 2, as proposed by [12]. In this system, the transmitter is represented as a Feedforward Neural Network (FNN) with dense layers and a normalization layer that is set to meet the physical constraints of the transmit vector x.

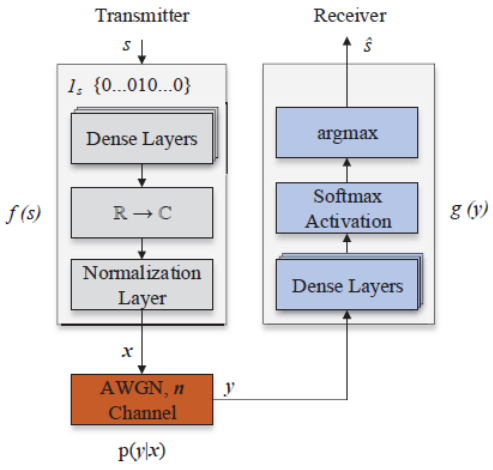

Fig. 2. Structure of a wireless communication system represented as an autoencoder.

As previously discussed, a conventional communication system is comprised of different blocks for channel coding/decoding and modulation/demodulation functions. In contrast, the AE-based system does not have explicit blocks but aims to optimize the system in an end-to-end process, which aligns with the system parameters such as the input message size, the number of channels uses per message, and transmit signal power constraints. These parameters are utilized to implement the autoencoder models, which are similar to the standard communication channel coded and uncoded communication systems, and their performance is compared over an AWGN channel. The input, encoder layer, channel, and decoder layer are represented as *s*, *enc(s)*, *cha(s)*, and *dec(s)*, respectively. The AE is trained using Adam (**Ada**ptive **m**oment estimation) optimizer to produce an output $de(cha(enc(s)))$ that minimizes an arbitrary loss function L (s, $dec(cha(enc(s)))$) [24].

The constellation diagrams produced by a single-antenna autoencoder system are not predetermined, but are instead learned based on the desired performance metric to be minimized at the receiver, such as symbol error rate, coherence time, distance, and propagation loss. The transmitter's hardware imposes specific limitations as outlined in reference [25]. The transmitter enforces the constraints mentioned below.

(a) An energy constraint $\|x\|_2^2 \leq n$,

(b) An amplitude constraint $|x_i| \leq 1\ \forall_i$,

(c) An average power constraint $E[|x_i|^2] \leq 1\ \forall_i$ on x.

The data rate of the system is calculated using the formula $R = k/n$ [bit/channel use]. The parameter k represents the number of input bits, and is equal to $log_2(M)$, where M is the number of possible messages that can be sent. The parameter n includes both the input bits and additional redundant bits used to reduce channel effects. The (n, k) notation indicates that the system sends one message from M possible messages (k bits) over n channel uses. Fig. 2 shows a block diagram of the channel autoencoder, which learns from the distribution of the communication channel data to compensate for impairments. The communication channel is defined by the density of the conditional probability $p(y|x)$, where $y \in R^n$ represents the signal at the receiver. The message is detected as y at the receiver, and the operation $r : R^n \rightarrow M$ is applied to estimate the value of the transmitted message s. The channel autoencoder is optimized to map x to y, which allows s to be recovered by minimizing the probability of error. The autoencoder components are summarized as follows: -

a) *Input*: The symbol s is transformed into a one-hot vector, meaning that it can only have valid combinations of values where one bit is set to '1' and all the others are set to '0'. This particular encoding enables a state machine to operate at a faster clock rate compared to other encodings. Moreover, determining the state of a one-hot vector requires accessing only one flip-flop, which has a low and consistent cost.

b) *Transmitter*: The transmitter consists of a feedforward neural network which has several dense layers. The output of the last dense layer is modified to represent two complex numbers for every modulated input symbol. These numbers represent the

real (in-phase, I) and imaginary (quadrature, Q) parts. The normalization layer is added to guarantee that the physical restrictions on x are satisfied.

c) *Channel*: The channel layer is fixed and cannot be trained. It is modelled as a layer of Additive White Gaussian Noise (AWGN) that is added to the signal. The variance of this noise is determined by a parameter $\beta = (\frac{2RE_b}{N_0})^{-1}$,which is calculated using the ratio $\frac{E_b}{N_0}$ of energy per bit (E_b) to noise power spectral density (N_0). The value of β changes for each training example, and the noise is only applied during the forward pass to simulate signal distortion, but it is not considered during the backward pass.

d) *Receiver*: similar to transmitter, it is constructed using a Fully connected Neural Network (FNN). Its final layer employs the softmax activation function to generate a probability vector $p \in (0, 1)^M$ representing all potential messages. The value in p with the greatest probability is designated as $\hat{s}$.

e) *Training*: The Adaptive Moment (Adam) optimizer is used to train the autoencoder and modify the weights of the FNN, and the performance is evaluated. The training batch comprises all potential messages $s \in M$.

4. Simulation Results and Performance Evaluation

The autoencoder operates by utilizing the data created during transmission and the identical data at the receiving end. As the data used is not externally labelled, the autoencoder is classified as an unsupervised learning system. This approach enables the autoencoder to acquire knowledge without any prior information. The input message is represented as a vector with only one element being "1" and the rest being "0." This is known as a one-hot vector. The channel through which the message passes is an Additive White Gaussian Noise (AWGN) channel. The AWGN channel adds noise to the message in order to achieve a specific energy per bit to noise power density ratio. In [26], researchers presented a (7,4) autoencoder network that utilizes energy normalization and has a training of 3 dB. To achieve optimal results with minimal complexity, both the encoder (transmitter) and the decoder (receiver) consist of two fully connected layers. The input layer (featureInputLayer) accepts a one-hot vector of length *M*. The first fully connected layer of the encoder has M inputs and M outputs, followed by a ReLU layer. The second fully connected layer has *M* inputs and n outputs, followed by a normalization layer. After the encoder layers, the AWGN channel layer is applied. The channel's output is then fed into the decoder layers, starting with a fully connected layer that has *n* inputs and *M* outputs, followed by a ReLU layer. The second fully connected layer has *M* inputs and *M* outputs, followed by a softmax layer (softmaxLayer), which produces the probability of each *M* symbol. Finally, the classification layer determines the most likely transmitted symbol from 0 to *M*-1.

A (2,2) autoencoder is trained using below specific parameters, including energy normalization.

- Adam (adaptive moment estimation) optimizer,
- Initial learning rate of 0.01,

- Maximum epochs of 15,
- Minibatch size of 20*M,
- Piecewise learning schedule with drop period of 10 and drop factor of 0.1.

The Adaptive Moment Estimation (Adam) optimizer algorithm can be used to train the parameters of the autoencoder to minimize the reconstruction error of the received signal. The Adam optimizer algorithm uses a combination of momentum and adaptive learning rate to converge to the minimum of the cost function efficiently. The cost function in the case of an autoencoder is the reconstruction error between the input and output signal.

During training, the Adam optimizer algorithm updates the parameters of the autoencoder using the following steps:

1. Initialize the first and second moment estimates:
   initial first moment vector $m_0 = 0$ and initial second moment vector $v_0 = 0$
2. Forward pass through the autoencoder to obtain the output signal.
3. Compute the reconstruction error between the input and output signal.
4. Compute the gradient of the cost function with respect to the parameters
   $g_t = \nabla_\theta J(\theta_t)$
5. Update the first moment estimate
   $m_t = \beta_1 * m_{t-1} + (1 - \beta_1) * g_t$
   where β_1 is the exponential decay rate for the first moment estimate, typically set to 0.9.
6. Update the second moment estimate:
   $v_t = \beta_2 * v_{t-1} + (1 - \beta_2) * g_t^2$
   where β_2 is the exponential decay rate for the second moment estimate, typically set to 0.999.
7. Compute the bias-corrected first moment estimate $\hat{m}_t$ and second moment estimate $\hat{v}_t$ using the following equations:

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t}$$

where t is the current iteration step, β_1 is the exponential decay rate for the first moment estimate, and β_2 is the exponential decay rate for the second moment estimate.

In addition to the standard hyperparameters used in the Adam optimizer, such as the learning rate and exponential decay rates for the first and second moment estimates, there are additional hyperparameters that are specific to the application of autoencoders for wireless communication. These include the Signal-to-Noise Ratio (SNR) and the batch size.

In Fig. 3, the training process at a noise level of 3dB has been illustrated, and it can be observed that the validation accuracy quickly surpasses 90%, while the validation loss consistently decreases. This pattern indicates that the initial training value was set low enough to produce some errors but not too low to prevent convergence.

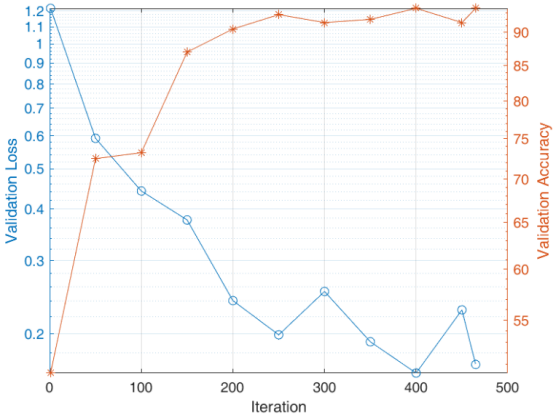

Fig. 3. The training process plot.

Fig.4 displays the layer diagrams of the complete autoencoder, including its encoder and decoder networks, represented by the objects generated from the trained network. The encoder network is also known as the transmitter, while the decoder network is known as the receiver.

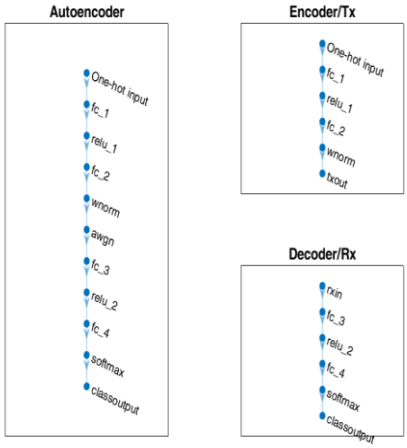

Fig. 4. The objects generated by the trained network.

The constellation learned by the autoencoder was plotted to send symbols through the AWGN channel together with the received constellation. For a (2,2) configuration, the autoencoder learned a QPSK ($M = 2^2 = 4$) constellation with a phase rotation. The received constellation was basically the activation values at the output of the channel layer obtained using the activation function and treated as interleaved complex numbers.

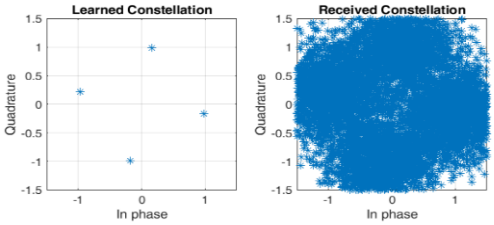

Fig. 5. The constellation diagrams produced by Autoencoder.

The Block Error Rate (BLER) performance of the (2,2) autoencoder was simulated by generating the random integers in the [0-1] range to represent random information bits. These information bits were then encoded into complex symbols. The real valued vector was mapped into a complex valued vector such that the odd and even elements were mapped into the in-phase and the quadrature component of a complex symbol, respectively. The array was treated as an interleaved complex array. The encoded complex symbols were passed through an AWGN channel to simulate channel impairment. The channel impaired complex symbols were then decoded and the simulation was run for each point for at least 10 block errors to compare the results with that of an uncoded QPSK system with block length 2.

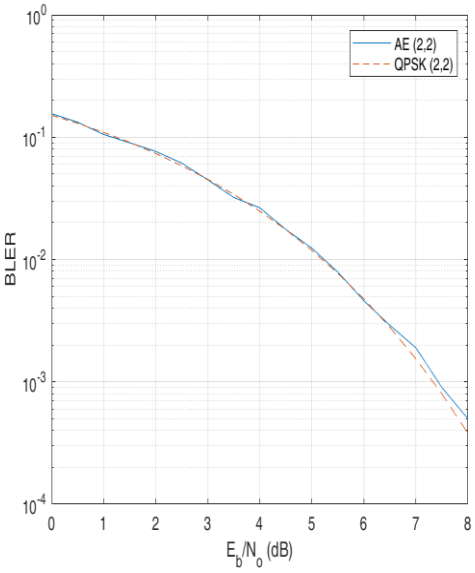

Fig. 6. The BLER plot for QPSK (2,2) and AE (2,2).

It can be inferred from Fig.6, indicating well-formed constellation together with the BLER results, that training for 15 epochs was enough to get a satisfactory

convergence. Learned constellations of several autoencoders normalized to unit energy and unit average power have also been generated and the same are shown in Fig.7. It is to be noted that Train (2,4) autoencoder was normalized to unit energy.

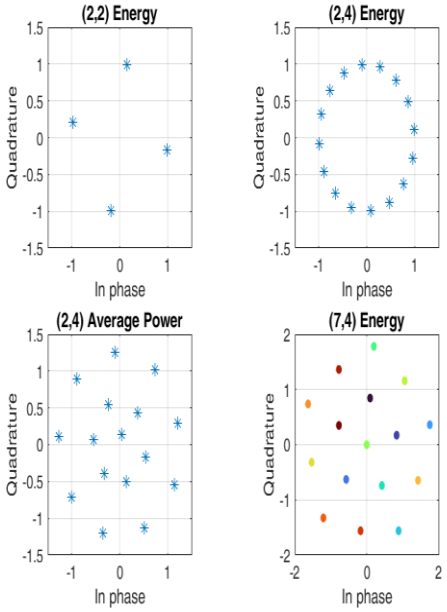

Fig.7. Comparisons of Constellation diagrams of several autoencoders.

The (2,2) autoencoder has been trained to reach a convergence point on a QPSK constellation, which has a phase shift that is optimal for the encountered channel conditions. On the other hand, the (2,4) autoencoder with energy normalization converges on a 16PSK constellation with a phase shift. It is important to note that energy normalization is used to ensure that every symbol has the same energy and is placed on the unit circle. Under this constraint, the best constellation is PSK constellation with equal angular distance between symbols. Finally, the (2,4) autoencoder with average power normalization converges to a three-tier constellation consisting of 1-6-9 symbols. The BLER performance of a (7,4) autoencoder was simulated with that of (7,4) Hamming code with QPSK modulation for both hard decision and maximum likelihood (ML) decoding. An uncoded (4,4) QPSK was used as a baseline. The (4,4) uncoded QPSK was essentially a PSK modulated system that sent blocks of 4 bits and measured BLER.

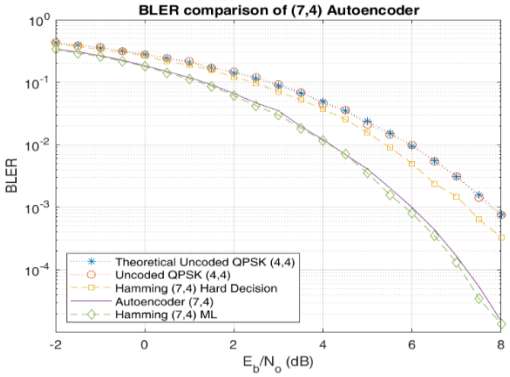

Fig. 8. (7, 4) Autoencoder BLER performance comparison.

Subsequently, the BLER performance of autoencoders with R=1 was simulated and compared with that of uncoded QPSK systems. The uncoded (2,2) and (8,8) QPSK were used as baselines. The BLER performance of these systems was compared with that of (2,2), (4,4) and (8,8) autoencoders.

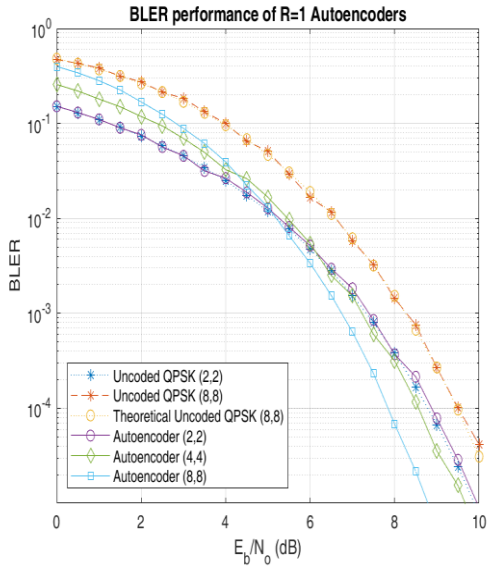

Fig. 9. Autoencoder of R = 1 BLER performance comparison.

The Bit error rate of QPSK was found to be the same for both (8,8) and (2,2) cases, however, the BLER was observed to be dependent on the block length, n, and became worse as n increased in accordance with relationship $BLER = 1 - (1 - BER)^n$. As expected, the BLER performance of (8,8) QPSK was observed to be worse than the (2,2) QPSK system. It was also observed that the BLER performance of (2,2) autoencoder matched the BLER performance of (2,2) QPSK. On the other hand, (4,4) and (8,8) autoencoders were found to optimize the channel coder and the constellation jointly in order to obtain a coding gain in comparison to the corresponding uncoded QPSK systems.

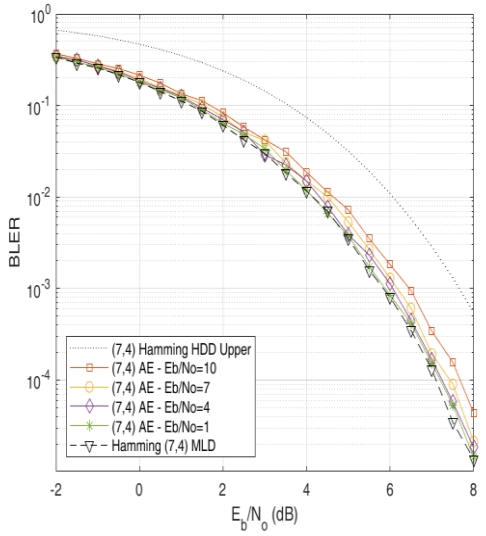

Fig.10. Autoencoder with Hamming code performance comparison.

The (7,4) autoencoder was trained with energy normalization under different values and the BLER performance has been compared. The BLER performance has also been plotted in Fig. 10, together with the theoretical upper bound for hard decision decoded Hamming (7,4) code and the simulated BLER of Maximum Likelihood Decoded (MLD) pertaining to Hamming (7,4) code. As the E_b/N_0 decreased from 10 dB to 1 dB, the BLER performance of the (7,4) autoencoder was observed to get closer to the Hamming (7,4) code with MLD, and at that point, it almost matched the MLD Hamming (7,4) code. This is a quite significant result as it establishes the possibility of learning the joint coding and modulation schemes by autoencoders in an unsupervised manner.

5. Conclusion and Future scope

In this paper, the use of deep learning architectures in optimizing communication systems has been brought out. The authors propose the implementation of an Autoencoder as a transmitter and receiver for the physical layer of communication. Instead of optimizing individual blocks of a conventional communication system, an end-to-end optimization approach has been suggested to minimize the reconstruction loss. The efficacy of this approach has also been demonstrated in capturing channel impairments in single antenna systems and matching modulation techniques using off-the-shelf DNNs. It has been concluded that autoencoders are capable of designing the end-to-end communication system in an unsupervised manner by learning the 'coding and modulation' as one entity.

With regards to the future work, multiple learning strategies can be explored on the autoencoder side, including different weight initialization, hyperparameter selection, and various emerging autoencoder architectures. Further, additional autoencoders can be utilized to extend this approach to multi-user and multiple-antenna systems. This work can also be applied to specific domains such as satellite communications, backhaul radios, dense urban wireless, 5G MIMO, amongst others.

References

1. E.A.A. Alaoui, S.C.K. Tekouabou, S. Hartini, Z. Rustam, H. Silkan, S. Agoujil, Improvement in automated diagnosis of soft tissues tumors using machine learning, Big Data Min. Anal. 4 (1) (2021) 33–46, http://dx.doi.org/10.26599/ BDMA.2020.9020023.
2. S. Shorewala, A. Ashfaque, R. Sidharth, U. Verma, Weed density and distribution estimation for precision agriculture using semi-supervised learning, IEEE Access 9 (2021) 27971–27986, http://dx.doi.org/10.1109/ACCESS.2021.3057912.
3. M. Varasteh, J. Hoydis, B. Clerckx, Learning to communicate and energize: Modulation, coding, and multiple access designs for wireless information-power transmission, IEEE Trans. Commun. 68 (11) (2020) 6822–6839, http://dx.doi.org/10.1109/TCOMM.2020.3017020.
4. J. Ren, G. Yu, G. Ding, Accelerating DNN training in wireless federated edge learning systems, IEEE J. Sel. Areas Commun. 39 (1) (2021) 219–232, http: //dx.doi.org/10.1109/JSAC.2020.3036971.
5. M.E. Morocho-Cayamcela, H. Lee, W. Lim, Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions, IEEE Access 7 (2019) 137184–137206.
6. M.E. Morocho Cayamcela, W. Lim, Artificial intelligence in 5G technology: A survey, in: 2018 International Conference on Information and Communication Technology Convergence (ICTC), 2018, pp. 860–865.
7. M.E. Morocho-Cayamcela, H. Lee, W. Lim, Machine learning to improve multihop searching and extended wireless reachability in V2x, IEEE Commun. Lett.(2020) 1.
8. J.N. Njoku, M.E. Morocho-Cayamcela, W. Lim, CGDNet: Efficient hybrid deep learning model for robust automatic modulation recognition, IEEE Netw. Lett. 3 (2) (2021) 47–51, http://dx.doi.org/10.1109/LNET.2021.3057637.
9. J.N. Njoku, M.E. Morocho-Cayamcela, W. Lim, Automatic radar waveform recognition using the wigner-ville distribution and AlexNet-SVM, in: Proceedings of the KICS Summer Conference, Pyeongchang, South Korea, 2020, pp. 1–4.
10. P. Popovski, A mathematical view on a communication channel, in: Wireless Connectivity: An Intuitive and Fundamental Guide, Wiley, Wiley, 2020, pp.145–173, http://dx.doi.org/10.1002/9781119114963.ch6.
11. F. Farzaneh, A. Fotowat, M. Kamarei, A. Nikoofard, M. Elmi, 1 the amazing world of wireless systems, in: Introduction to Wireless Communication Circuits, River Publishers, 2020, pp. 1–26.
12. T. O'Shea, J. Hoydis, An introduction to deep learning for the physical layer, IEEE Trans. Cogn. Commun. Netw. 3 (4) (2017) 563–575.
13. T.J. O'Shea, K. Karra, T.C. Clancy, learning to communicate: Channel autoencoders, domain specific regularizers, and attention, in: 2016 IEEE International Symposium on Signal Processing and Information

Technology, ISSPIT 2016, 2017, pp. 1–6.

14. H. Zhang, L. Zhang, Y. Jiang, Overfitting and underfitting analysis for deep learning based end-to-end communication systems, in: 2019 11th International Conference on Wireless Communications and Signal Processing (WCSP), 2019, pp. 1–6, http://dx.doi.org/10.1109/WCSP.2019.8927876.
15. L. Liu, T. Lin, Y. Zhou, A deep learning method-based receiver design, in: 2020 IEEE 6th International Conference on Computer and Communications (ICCC), 2020, pp. 975–979, http://dx.doi.org/10.1109/ICCC51575.2020.9344965.
16. L. Liu, Y. Luo, X. Shen, M. Sun, B. Li, β -Dropout: A unified dropout, IEEE Access 7 (2019) 36140–36153, http://dx.doi.org/10.1109/ACCESS.2019.2904881.
17. M. Soltani, W. Fatnassi, A. Aboutaleb, Z. Rezki, A. Bhuyan, P. Titus, Autoencoder based optical wireless communications systems, in: 2018 IEEE Globecom Workshops, GC Wkshps 2018 - Proceedings, Institute of Electrical and Electronics Engineers Inc., 2019, pp. 1–6, http://dx.doi.org/10.1109/GLOCOMW.2018.8644104.
18. T.J. O'Shea, T. Erpek, T.C. Clancy, Physical layer deep learning of encodings for the MIMO fading channel, in: 55th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2017, 2018-January, Institute of Electrical and Electronics Engineers Inc., 2017, pp. 76–80, http://dx.doi.org/10.1109/ALLERTON.2017.8262721.
19. T.J. O'Shea, T. Roy, N. West, B.C. Hilburn, Physical layer communications system design over-the-air using adversarial networks, in: European Signal Processing Conference, 2018-Septe, 2018, pp. 529–532, http://dx.doi.org/10.23919/EUSIPCO.2018.8553233.
20. M.E. Morocho Cayamcela, J.N. Njoku, J. Park, W. Lim, Learning to communicate with autoencoders: Rethinking wireless systems with deep learning, in: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Fukuoka, Japan, 2020, pp. 308–311.
21. S. Mohamed, J. Dong, A.R. Junejo, D.C. Zuo, Model-based: End-to-end molecular communication system through deep reinforcement learning auto encoder, IEEE Access 7 (2019) 70279–70286.
22. H. Lee, S.H. Lee, T.Q.S. Quek, I. Lee, Deep learning framework for wireless systems: Applications to optical wireless communications, IEEE Commun. Mag. 57 (3) (2019) 35–41.
23. D. Wu, M. Nekovee, Y. Wang, Deep learning-based autoencoder for m-user wireless interference channel physical layer design, IEEE Access 8 (2020) 174679–174691, http://dx.doi.org/10.1109/ACCESS.2020.3025597.
24. D.J. Ji, J. Park, D.-H. Cho, ConvAE: A new channel autoencoder based on convolutional layers and residual connections, IEEE Commun. Lett. 23 (10) (2019) 1769–1772, http://dx.doi.org/10.1109/lcomm.2019.2930287.

25. T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," in IEEE Transactions on Cognitive Communications and Networking, vol. 3, no. 4, pp. 563-575, Dec. 2017, doi: 10.1109/TCCN.2017.275837.
26. T. J. O'Shea, T. Erpek, and T. C. Clancy, "Physical layer deep learning of encodings for the MIMO fading channel," in 2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton). IEEE, 10 2017, pp. 76–80.