Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

LDPC CODE BASED AUTOENCODER OF AWSN USING DEEP NEURAL NETWORKS MODEL FOR WIRELESS COMMUNICATION CHANNEL

VAISSNAVE V
Department of Computational Intelligence, SRM Institute of Science and Technology, Kattankulathur, India.
Email: Vaissnave@gmail.com

AMUTHACHENTHIRU K
Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Virudhunagar, India.
Email: kamuthachenthiru@gmail.com

DURGA DEVI G
Department of Computer Science and Engineering, Thamirabharani Engineering College, Tirunelveli, India.
Emai: ldurgaaganesan@gmail.com

Dr. ANNA LAKSHMI A
Department of Information Technology, RMK Engineering College, Chennai, India.
Email: annalaxmi.raj@gmail.com

Dr. JENIFER JOHN J
Assistant Professor, Department of Electronics and Communication Engineering, Jayaraj Annapackiam CSI College of Engineering, Nazareth, Tamilnadu. Email: johnjjesun@gmail.com

RAMNATH M
Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Virudhunagar, India.
Email: ramnath25@gmail.com, ORCID ID: 0000-0002-9397-9320

Dr. MARIAPPAN E
Department of Artificial Intelligence and Data Science, Ramco Institute of Technology, Virudhunagar, India.
Email: mapcse.e81@gmail.com

Abstract

A wireless communication system employs a variety of data compression, encoding, and modulation techniques to efficiently transmit messages via communication channels, aiming to reproduce the information at the receiver with the fewest possible errors. To counteract the impact of noise and interference encountered by the signal during its journey through the communication channel, the channel encoder introduces redundancy to the binary information sequence. The channel decoder at the receiver utilizes this redundancy to combat errors. To enhance data redundancy, the channel encoder utilizes error-correcting codes, including block codes, convolutional codes, Low-Density Parity Check (LDPC) codes, and turbo codes. These coding methods play a crucial role in error detection and correction. However, the configuration of a wireless communication system can now be simplified by leveraging Deep Neural Networks (DNNs). This streamlined communication system can be conceptualized as a specific type of autoencoder in the realm of Deep Learning (DL). The primary goal of this research is to develop an autoencoder model for Additive White Gaussian Noise (AWGN) and fading channels with a low error probability, ensuring reliable communication.

Keywords: Compression, Interference, Low-Density Parity Check (LDPC), Deep Neural Networks (DNNs), Additive White Gaussian Noise (AWGN).

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

I. INTRODUCTION

Error-correcting block codes divide data into discrete blocks. Each block of n bits (where n > k) is mapped to a corresponding block of k message bits. The transmitter adds n-k redundant bits to facilitate error detection and correction. The coding rate is specified as the k/n ratio, and the code is denoted as an (n, k) block code. In a linear block code, the resulting codeword is a linear combination of any two code words. Parity bits and message bits in linear block codes are subjected to a linear combination.

Let A be the input channel. White additive Gaussian noise, called N, was used. Subsequently, the channel output is.

$$Y = A + N \qquad \text{-- (1)}$$

Fading in wireless communication refers to fluctuations in signal attenuation under various circumstances. These fluctuations can be caused by factors such as time, shifts in geographic location, and changes in radio frequency. We are modelling the fading channel for this project as

$$Y = A * H + N \qquad \text{-- (2)}$$

Where Y is the channel output, A is its input and H and N are random numbers.

H is a number with a mean of 0 and standard deviation of 1 from the complex random Gaussian distribution.

There are two forms of detection, both based on Channel State Information (CSI):

CSI can be determined if the H value is known. The phase of the received symbols is rectified using H, and the sent bits are then retrieved using a maximum-likelihood decoder. Anticipating that the network will learn phase correction during decoder training, we pass not only the received bits but also the CSI to the decoder network. This type of detection is employed when the CSI is unknown or does not match the precise channel-gain value. To train the decoder, we send the received bits, channel gain, and a random number (introducing an error) to the decoder network. When compared to traditional optimal statistical inference techniques, a trained DNN requires significantly less computational complexity for executing inference tasks. Additionally, while the majority of current ECCs are linear, DNNs are nonlinear. Therefore, utilizing DNNs can result in nonlinear codes that may be more effective than linear codes.

II. RELATED WORKS

An unsupervised artificial neural network, called an autoencoder, learns to efficiently compress and encode data before learning to reconstruct the data from the compressed, encoded representation to a form that is as similar to the original input as possible. The autoencoder can nonlinearly compress and reconstruct the input in this manner. In our case, the autoencoder serves a different purpose. To retrieve the transmitted message with a low likelihood of error, it aims to develop representations of messages that are resilient to channel impairments, mapping x to y. In contrast to other autoencoders, which

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

typically add redundancy to input data to increase compression, this autoencoder frequently subtracts redundancy to develop an intermediate representation that is resistant to channel perturbations

In the context of communication systems, the auto-encoder model appears as follows:

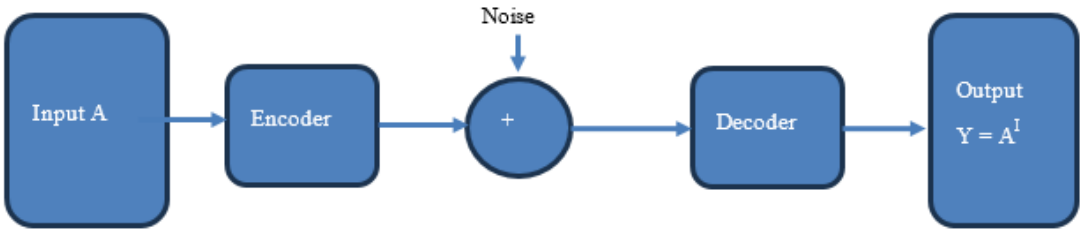

Fig 1: Autoencoder Model

According to a proposal made by Morocho-Cayamcela et al., the established communication system should have a transmitter, receiver, and channel that transfer information between them. By training them as DNNs, deep learning (DL) can jointly optimize numerous communication blocks at the transmitter and receiver, as compared to the separate block optimization of conventional communication systems. The output constellation diagrams in an AE system are learned based on the intended performance measure to be minimized at the receiver, rather than being predefined [1]. In the AE structure, a single DNN, referred to as the encoder, replaces the entire transmitter, with the input being the original bit information and the output being the broadcast signal. Another DNN, referred to as the decoder, takes the received signal as input and produces the reconstructed bits, thereby replacing the receiver. The encoder and decoder networks were concurrently trained by reducing the difference in loss between the original and reconstructed information. The issue of gradient backpropagation in practical wireless channels can also be addressed using GANs and retraining techniques [2].

Gradient descent and its variations can be used to train the TurboAE from start to finish. A block length of 100 was used for training and testing the CNN-AE and TurboAE. While TurboAE and Turbo codes show decreasing error rates as the block length increases, CNN-AE exhibits a saturating block length gain as the block length increases. CNNs or other general-purpose neural networks cannot perform well on extended block lengths when naively applied to a channel coding task [3]. A deep autoencoder framework can be considered an end-to-end communication paradigm. Even if the channels are unknown, gradients can be back propagated to the transmitter using conditional GAN. The Channel GAN Training Algorithm serves as the foundation for the channel GAN training process. The generator and discriminator are iteratively trained during each cycle, with the parameters of one model fixed while training the other. Using the learned transmitter, the encoded signal from the transmitter traveling via the actual channel can be used to retrieve the true data, whereas the encoded signal traveling through the channel generator is used to obtain the false data. The loss function in the equations is employed to update the parameters of the generator and discriminator [4].

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

In the proposed configuration, the first neural network represents or encodes an input signal with high-dimensional output, after which channel distortions are added. The second neural network then maps this high-dimensional signal to the input signal space. The k-length message serves as the input for the autoencoder, producing vectors with a length of 2k that are identical to the input vectors. Both the encoder and decoder networks consist of several fully connected layers. It is evident that the suggested DNN-trained codes function nearly as well as the theoretically ideal block codes with MDD in an AWGN channel [5]. To train an end-to-end transceiver in situations with time-varying channels, Ge et al. presented an MPA-AE structure and its accompanying algorithms. When the channel changes from the one that the transceiver is trained with, the MPA layer placed between the encoder and decoder of the classic AE can be fine-tuned. Simulations demonstrated the performance of the proposed method [6].

In a variety of use cases, such as single-user and multiuser scenarios, the MPA layer can be adaptably merged with the AE-based transceiver. Source coding schemes, high-order modulation schemes, huge MIMO schemes, and pre-distortion schemes, which can adapt well to time-varying channel circumstances, can all be designed using the MPA layer in single-user scenarios. According to theory, a transmitter's DNN and MPA layers should skew the input distribution to match the existing channel-distortion distribution. In terms of Bit Error Rate (BER), the proposed End-to-End (E2E) wireless communication system based on the DNN channel module performed comparably to the conventional communication system and the E2E communication system using Conditional Generative Adversarial Network (CGAN) as the unknown channel, with only a slight performance advantage. Our approach performs marginally worse than the conventional communication system in terms of Block Error Rate (BLER) but marginally better than the E2E communication system using CGAN as an unidentified channel [7]. Sandesh et al. proposed an architecture that utilizes an FCDNN in conjunction with a pre-processing layer that applies Channel State Information (CSI) knowledge to perform operations on the received data. The real and imaginary components of the pre-processed data were combined and supplied to the network as inputs. The output of the decoder network is a one-hot vector [8].

The binary cross-entropy loss for various insights into the flat fading complex channel at decreasing Signal-to-Noise Ratio (SNR) was minimized to train this network. A unique RNN-based neural network architecture that meets the low-latency criteria was proposed by Yihan et al. for LEARN. The causal neural encoder consists of a Fully Connected Neural Network (FCNN) with two layers of GRU added to a causal RNN. The neural structure guarantees the ability to learn and extend the best temporal storage to a nonlinear regime embedded in the neural structure [9]. Hinton et al. proposed an autoencoder, typically used to obtain a lower-dimensional representation of the input data. Autoencoders are composed of two neural networks connected to the back. In traditional autoencoders, the input data are represented or encoded by the first neural network into a smaller dimensional output, and the original data are recovered by the second neural network decoding the compressed data [10]. According to the setup proposed by David et al., the input signal is encoded or represented with high-dimensional

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

output by the first neural network. Channel distortions were then applied, and the second neural network mapped this high-dimensional signal to the input signal space. The k-length message was the input of the autoencoder, which produces one-hot vectors of length 2k as the output [11], matching the input vectors. The networks for the encoder and decoder were a series of fully connected layers.

III. PROPOSED METHODOLOGY AND RESULTS

3. Deep Learning based Block codes

The effectiveness of several types of autoencoders has been thoroughly assessed across various channels. Block codes are represented by the pair (n, k), where k represents the message length, and n is the code word length.

The autoencoder was trained as described below, and the encoder neural network receives 2k potential messages in a random order. The ReLU and tanh activation functions were used to construct the encoder neural network layers. The wireless channel receives the output of the encoder neural network. The decoder neural network receives the input from the output of the wireless channel. This training is performed with decreasing levels of signal-to-noise ratio for several instances of CN (0, 1) random variables. The autoencoder was trained through two steps of weight optimization. The first-stage optimization of the autoencoder is solely used to update the weights of the encoder network to maximize the distance separation of the code words.

The loss function can be written as

$$L_e(\Theta_e) = - \lambda * d_{min}(\Theta_e) \qquad \text{-- (3)}$$

Θ_e stands for the encoder network outputs, and θ is the regularization parameter used for optimizing the codewords. The minimization of this loss results in the maximization of d_{min}, as the negative sign of the loss function ensures this. The root mean squared error was used as the loss function in the second-stage optimization, which updated the weights of the decoder network.

3.1. (7, 4) Auto-encoder

To increase the distance between the code words, the encoder network is trained using 32 message bits that may be employed. Then, an autoencoder model is trained with the introduction of various channel types, utilizing the weights of the encoder. The channel receives the encoder's outputs or 7-bit code words, and the decoder network receives code words from the channel. The decoder network is trained to recover the 4-bit message from the code words.

3.1.1. Architecture of Auto-Encoder

Fully-connected neural networks (FCNN), also known as dense layers, serve as the building blocks of encoder networks. These layers are then activated through functions such as ReLU and tanh. In the encoder, the last layer is activated by the tanh function to mimic QAM modulation, while all other layers are activated by the ReLU function.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

The decoder network is similar, consisting of dense layers and being fully activated by ReLU/Leaky ReLU activations. The input dimensions of the decoder network vary depending on the channel.

Table 1: consists of the encoder architecture and Table 3.2 has the required hyperparameters to train the network

Component	Layer	Activation Function	Layer Dimensions
Encoder	Dense	ReLU	4, 32
	Batch Normalisation		32
	Dense	ReLU	32, 16
	Batch Normalisation		16
	Dense	tanh	16,7
	Layer Normalisation		7

Table 2: Encoder Architecture for (7,4) Auto-encoder

Loss Function: - $\lambda * d_{min}(\Theta_e)$
Optimizer: Adam
Batch Size: 16
Epochs: 30000
λ: 1

The highest squared Euclidean distance between the codewords for a (7,4) encoder obtained after training for 30,000 epochs is 10.05, whereas the theoretical squared Euclidean distance is 12. We now train the decoder in several channels using these encoder weights, and we assess the Bit Error Rate (BER) vs Signal-to-Noise Ratio (SNR) curves.

3.1.2 AWGN channel

We can directly feed the received bits to the decoder network and train it to learn how to recover the message bits since the Additive White Gaussian Noise (AWGN) channel only introduces additive noise, and that noise affects only the real portion of the signal. Batch normalization is not applied to the (16,8) layer because it is only applied to layers with larger dimensions.

Table 3: Decoder Architecture (7,4) Auto-encoder in AWGN channel

Component	Layer	Activation Function	Layer Dimensions
Channel	AWGN		7
Decoder	Dense	ReLU	4, 32
	Batch Normalisation		32
	Dense	ReLU	32, 16
	Batch Normalisation		16
	Dense	ReLU	16,8
	Dense	ReLU	8, 4

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

We can assess the entire autoencoder model after training the decoder. The Bit Error Rate (BER) vs. Signal-to-Noise Ratio (SNR) curves for the (7,4) Autoencoder model are shown in Fig. 4.7. It is evident that the (7,4) Hamming code and the (7,4) Autoencoder differ by 2 dB. This issue has arisen since the encoder model could only attain a distance of 10.05 between codewords. This issue might be resolved if we attempt to achieve a theoretically attainable distance.

Table 4: Hyper Parameters for (7,4) decoder

Loss Function: MMSE
Optimizer: Adam
Batch Size: 1024
Epochs: 10000

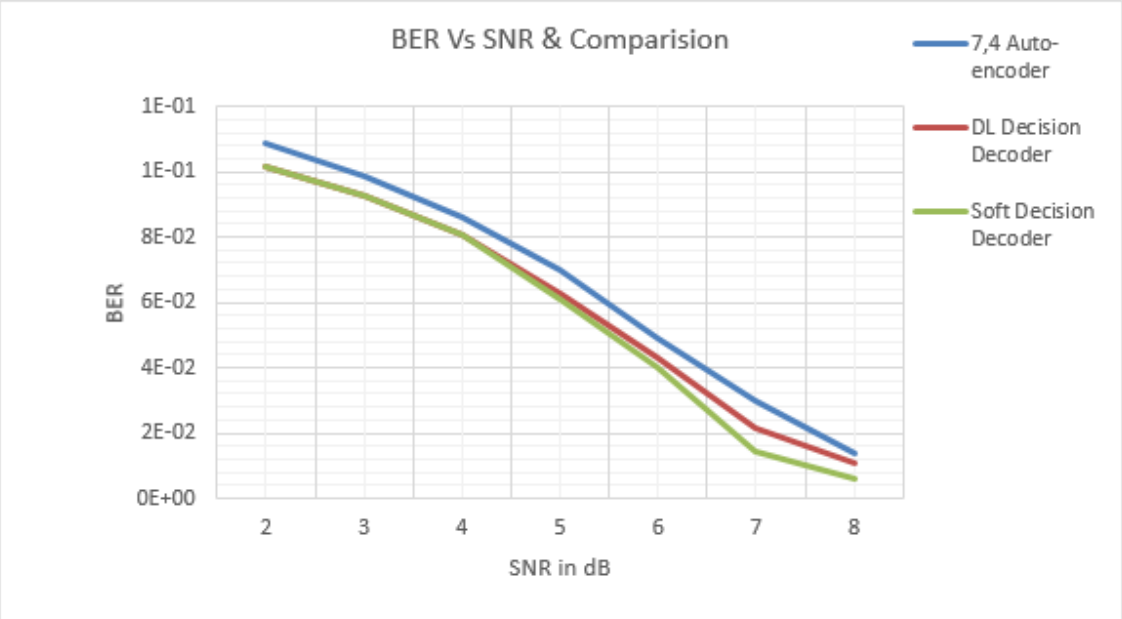

Fig 2: (7,4) Auto-encoder in AWGN channel

3.1.3 Fading channel with exact CSI

The output of the encoder experiences multiplicative and additive noise due to the fading channel. Although the encoder outputs are real numbers, when they are sent across the fading channel, the decoder receives both real and imaginary components. The exact h value, along with the real and imaginary components, is concatenated and provided as input to the decoder network.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

Table 3.5: consists of the decoder architecture, and the hyperparameters needed for training are the same as in Table 3.4

Component	Layer	Activation Function	Layer Dimensions
Channel	Fading (h)		1
	Additive noise (n)		7
Decoder	Dense	ReLU	4, 32
	Batch Normalisation		32
	Dense	ReLU	32, 16
	Batch Normalisation		16
	Dense	ReLU	16,8
	Dense	ReLU	8, 4

Following Fig. 3 are the BER vs SNR curves for the (7,4) Autoencoder model. We can see that there is a 2-5 dB gap between the (7,4) hamming code and (7,4) Autoencoder.

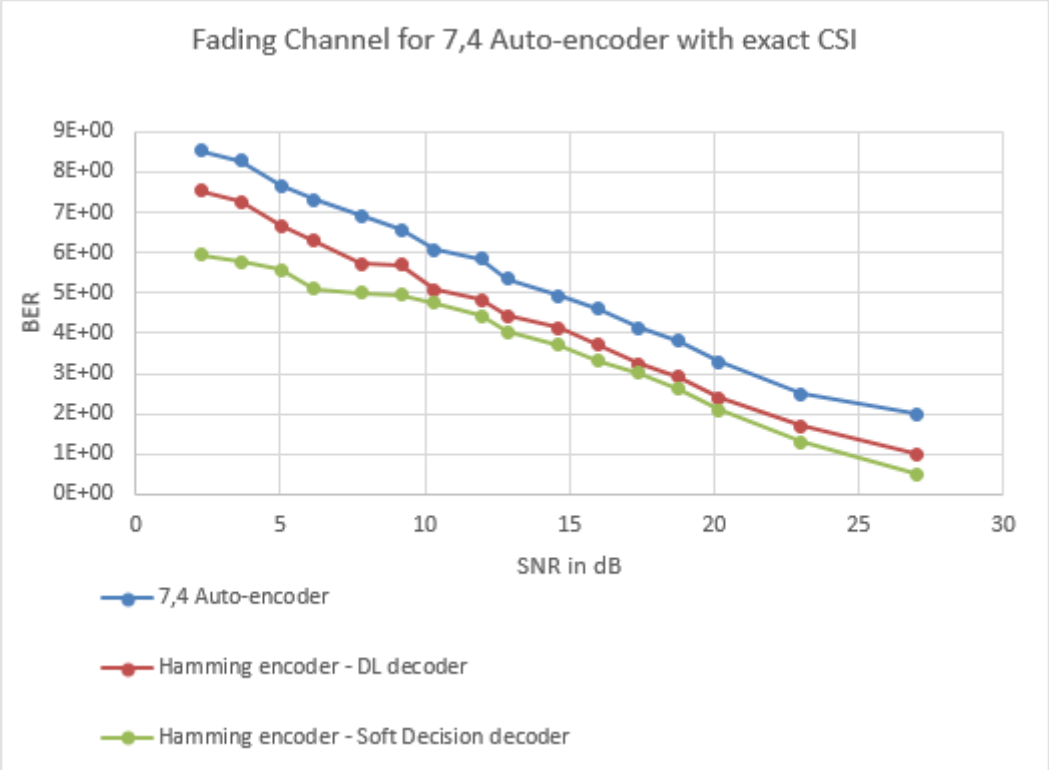

Fig 3: are the BER vs SNR curves for the (7,4) Autoencoder model

3.2 (15,11) Auto-encoder

To increase the distance between code words, the encoder network was trained with 512 potential message bits. Then, an autoencoder model is trained with the introduction of various channel types, utilizing the weights of the encoder. The channel receives the encoder's outputs, or 15-bit code words, and the decoder network receives code words

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

from the channel. The decoder network was trained to extract the 11-bit message from the code words.

3.2.1 Architecture of Auto-Encoder

Fully-connected neural networks (FCNN), also known as dense layers, serve as the building blocks of encoder networks. These layers are then activated by means of activation functions such as ReLU and tanh. In the encoder, the last layer is activated by the tanh function to mimic BPSK modulation, whereas all other layers are activated by the ReLU function. The decoder network is similar, consisting of dense layers, and ReLU/Leaky ReLU activations fully activate it. The input dimensions of the decoder network vary depending on the channel. Therefore, only the encoder architecture is described in this section, and the decoder architecture is presented in depth in the channel sections where it belongs. Table 6 consists of the encoder architecture and Table 7 has the required hyperparameters to train the network.

Table 6: Encoder Architecture for (15,11) Auto-encoder

Component	Layer	Activation Function	Layer Dimensions
Encoder	Dense	ReLU	11, 64
	Batch Normalisation		64
	Dense	ReLU	64, 32
	Batch Normalisation		32
	Dense	tanh	32, 15
	Layer Normalisation		15

Table 7: Hyper Parameters for (15,11) encoder

Loss Function: $- \lambda * d_{min}(\Theta_e)$
Optimizer: Adam
Batch Size: 2048
Epochs: 40000
λ: 1

The highest squared Euclidean distance between the codewords for a (15,11) encoder obtained after training for 40,000 epochs was 9.46, whereas the theoretical squared Euclidean distance was 12. We now train the decoder in several channels using these encoder weights and assess the Bit Error Rate (BER) vs. Signal-to-Noise Ratio (SNR) curves.

3.2.2 AWGN channel

We can directly feed the received bits to the decoder network and train it to extract the message bits because the Additive White Gaussian Noise (AWGN) channel only introduces additive noise, which affects only the real portion of the signal. Table 8 consists of the decoder architecture and Table 9 has the required hyperparameters to train the network.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

Table 8: Decoder Architecture for (15,11) Auto-encoder in AWGN channel

Component	Layer	Activation Function	Layer Dimensions
Channel	AWGN		15
Decoder	Dense	ReLU	15, 64
	Batch Normalisation		64
	Dense	ReLU	64, 32
	Batch Normalisation		32
	Dense	ReLU	32
	Dense	ReLU	32, 11

We can assess the entire autoencoder model after training the decoder. The Bit Error Rate (BER) vs. Signal-to-Noise Ratio (SNR) curves for the (15,11) Autoencoder model are shown in Fig. 3.3. It is evident that the (15,11) Hamming code and the (15,11) Autoencoder differ by 3–5 db.

Table 9: Hyper Parameters for (15,11) decoder

Loss Function: MSE
Optimizer: Adam
Batch Size: 2048
Epochs: 5000

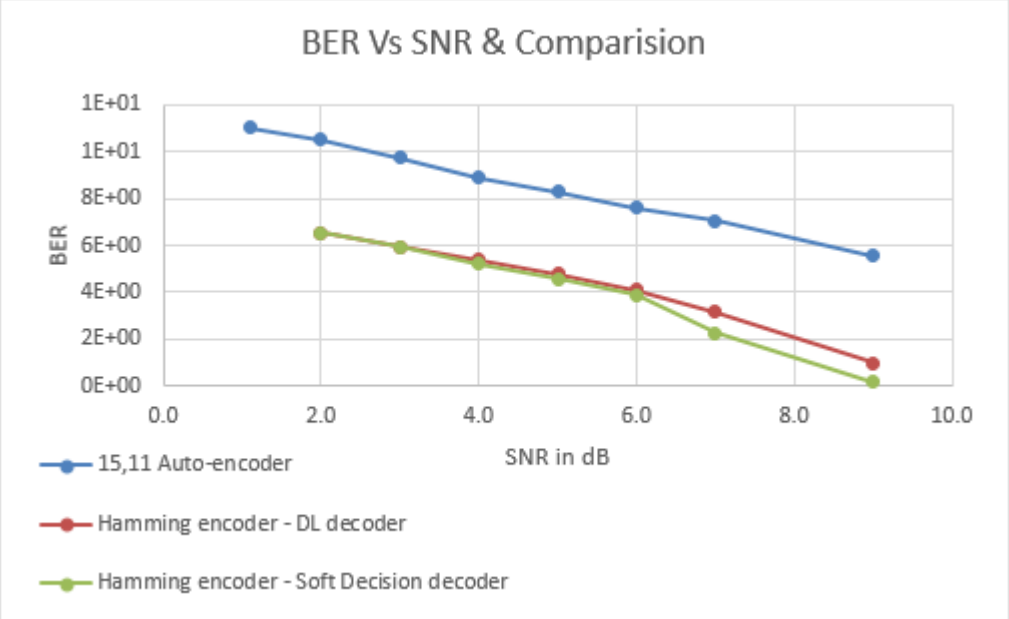

Fig 4: (15,11) Auto-encoder in AWGN channel

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

3.2.3 Fading channel with exact CSI

The output of the encoder experiences both multiplicative and additive noises owing to the fading channel. Although the encoder outputs are real numbers, the decoder receives both real and imaginary components when they are sent across the fading channel. The exact h value, along with the real and imaginary components, is combined and provided as input to the decoder network.

Table 10: consists of the decoder architecture and the hyperparameters needed for training are the same as in Table 9

Component	Layer	Activation Function	Layer Dimensions
Channel	Fading (h)		1
	Additive noise (n)		15
Decoder	Dense	ReLU	31, 128
	Batch Normalisation		128
	Dense	ReLU	128, 64
	Batch Normalisation		64
	Dense	ReLU	64, 32
	Batch Normalisation		32
	Dense	ReLU	32, 11

Table 10 presents the Decoder Architecture for the (15,11) Autoencoder in a fading channel with exact Channel State Information (CSI). Following Fig. 3.4 are the Bit Error Rate (BER) vs. Signal-to-Noise Ratio (SNR) curves for the (15,11) Autoencoder model. It is evident that there is a 2-5 dB gap between the (15,11) Hamming code and the (15,11) Autoencoder.

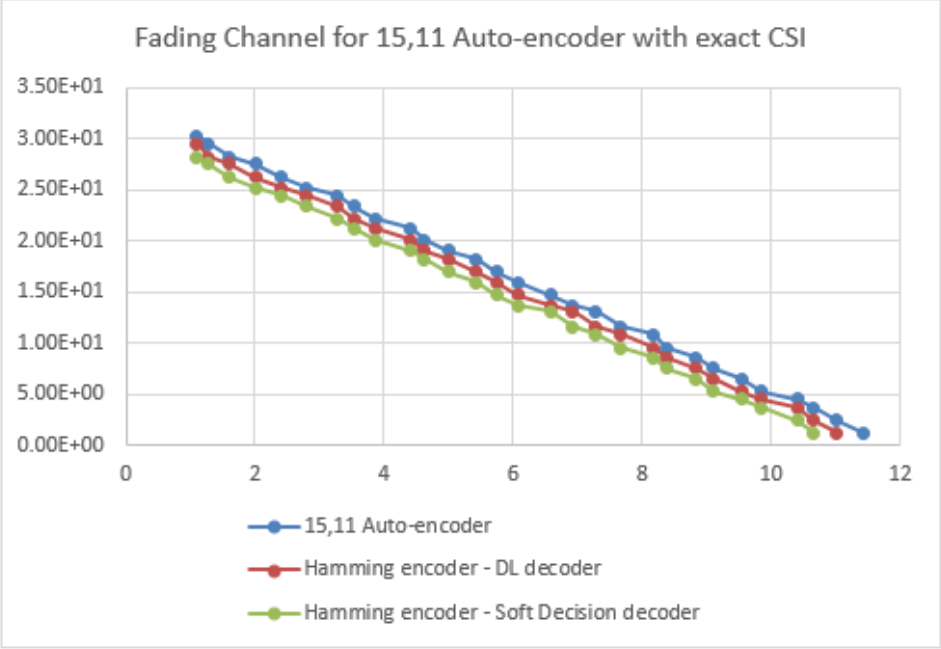

Fig 5: (15,11) Auto-encoder in Fading channel with exact CSI

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

IV. CONCLUSION

For wireless channels, such as AWGN and fading channels, we trained neural networks to learn nonlinear error control codes, and we obtained respectable error probabilities. Using deep-learning techniques, we created block codes for several wireless channels, such as the AWGN channel. We modelled a loss function that maximizes the distance between code words while implementing block codes to train the encoder. Hamming distance has been utilized extensively in research to train encoders, but we tested the effectiveness of the squared Euclidean distance loss instead.

Conflicts of Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declarations

Ethical Approval: Our work does not require ethical approval as it does not involve human subjects or sensitive data.

Competing Interests: There are no competing interests associated with our work.

Funding: We did not receive any funding for this research.

Availability of Data and Materials: The dataset used in this study was sourced from Kaggle and will be made available as an open-source resource. If the reviewer requests access to the data, we will gladly provide it.

References

1) Manuel Eugenio Morocho-Cayamcela and Wansu Lim, “Accelerating Wireless Channel Autoencoders for Short Coherence-time Communications”, Journal of Communications and Networks, Volume 22, No. 3, June 2020.

2) Siqi Liu, Tianyu Wang, Shaowei Wang, “Toward intelligent wireless communications: Deep learning - based physical layer technologies”, Digital Communications and Networks, Volume 7, October 2021.

3) Yihan Jiang, Hyeji Kim, Himanshu Asnani, Sreeram Kannan, Sewoong Oh and Pramod Viswanath, “Turbo Autoencoder: Deep learning-based channel codes for point-to-point communication channels”, 33rd Conference on Neural Information Processing Systems, 2019.

4) Hao Ye, Le Liang, Geoffrey Ye Liand Biing-Hwang Juang, “Deep Learning-Based End-to-End Wireless Communication Systems with Conditional GANs as Unknown Channels”, IEEE Transactions on Wireless Communications, Vol. 19, No. 5, May 2020.

5) Sandesh Rao Mattu, Lakshmi Narasimhan T and A. Chockalingam, “Autoencoder based Robust Transceivers for Fading Channels using Deep Neural Networks”, IEEE 91st Vehicular Technology Conference, June 2020.

6) Yiqun Ge, Wuxian Shi, Jian Wang, Rong L and Wen Tong “Joint Message Passing and Auto-Encoder for Deep Learning”, CEUR Workshop Proceedings, July 2022.

7) Yongli An, Shaomeng Wang, Li Zhao, Zhanlin Ji, Máirtín O’droma and Ivan Ganchev, “End-to-End Wireless Communication System Based on Deep Neural Network Channel Module”, IEEE Access, 2022.

Tianjin Daxue Xuebao (Ziran Kexue yu Gongcheng Jishu Ban)/
Journal of Tianjin University Science and Technology
ISSN (Online):0493-2137
E-Publication: Online Open Access
Vol: 57 Issue: 11:2024
DOI: 10.5281/zenodo.14043365

8) L. N. T. Sandesh Rao Mattu and A. Chockalingam, “Autoencoder based robust transceivers for fading channels using deep neural networks," IEEE 91st Vehicular Technology Conference, 2020.

9) H. A. Yihan Jiang, Hyeji Kim, “Learn codes: Inventing low-latency codes via recurrent neural networks," 53rd IEEE International Conference on Communications 2019.

10) G. E. Hinton and R. S. Zemel, “Autoencoders, minimum description length and Helmholtz free energy,” Proc. NIPS, pp. 3-10, Nov. 1993.

11) H. David and H. Sarah, “Digital Design and Computer Architecture”, 2007.

12) C. Jiang, H. Zhang, Y. Ren, Z. Han, K. C. Chen, and L. Hanzo, “Machine learning paradigms for next-generation wireless networks,” IEEE Wireless Commun., vol. 24, no. 2, pp. 98–105, Apr. 2017.

13) M. E. Morocho-Cayamcela, H. Lee, and W. Lim, “Machine learning to improve multi-hop searching and extended wireless reachability in V2X,” IEEE Commun. Lett., early access, 2020.

14) E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be’ery, “Deep learning methods for improved decoding of linear codes,” IEEE Journal of Selected Topics in Signal Processing, 2018.

15) Y. Jiang, H. Kim, S. Kannan, S. Oh, and P. Viswanath, “Deepturbo: Deep turbo decoder,” 2019.

16) A. Felix, S. Cammerer, S. Dörner, J. Hoydis, and S. t. Brink, “Ofdm-autoencoder for end-to-end learning of communications systems,” arXiv preprint arXiv:1803.05815, 2018.