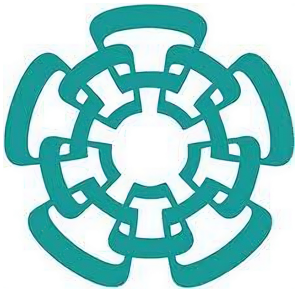

Centro de Investigación y de Estudios Avanzados del I.P.N

Unidad Guadalajara

Exploring Deep Learning Techniques for Equalizing Doubly Dispersive Channels

A thesis presented by:

Luis Emilio Tonix Gleason

to obtain the degree of:

Master in Science

in the subject of:

Electrical Engineering

Thesis Advisors:

Dr. Ramón Parra Michel

Dr. Fernando Peña Campos

Guadalajara, Jalisco January 13, 2023

Acknowledgment

Thank you for reading this; I appreciate it. I'm expecting that your work will benefit greatly from this material. I won't hesitate to say that you might have done your job more effectively than I did. Progress in science is a way to skepticism and having the best information from multiple sources to meet your individual standards.

Resumen

En la actualidad, la tecnología de redes de comunicación inalámbricas de alta movilidad está en constante evolución, y tiene aplicaciones en diversos ámbitos, tales como la seguridad vial, la conducción autónoma, el monitoreo remoto de vehículos, el vuelo de drones y los requisitos de 6G. Los transmisores RF, que se emplean en los sistemas de telecomunicaciones para transmitir información, codifican dicha información a través de la amplitud y la fase de una señal portadora, comúnmente conocida como datos IQ. Sin embargo, al viajar a través de un canal de alta movilidad, estos datos sufren distorsión, debido a la dispersión doble de dicho canal, lo que implica que las propiedades de la señal se ven afectadas por el desplazamiento, difusión en el tiempo y la frecuencia. Aunque existen algoritmos tradicionales que requieren ecualizadores iterativos complejos, que pueden ser lentos en términos de tiempo de ejecución pero precisos, también existen ecualizadores más sencillos, aunque menos precisos. En este trabajo se analizarán algunas soluciones basadas en redes neuronales, buscando un equilibrio entre la complejidad del tiempo y la precisión para hacer frente a este desafío.

Es posible afirmar que los algoritmos convencionales han demostrado ser eficaces en el desarrollo actual. Sin embargo, los recientes avances en redes neuronales han simplificado problemas que antes eran intratables, como el descifrado de secuencias de ADN o la creación de imágenes a partir de texto. En resumen, abordan con éxito problemas no lineales. El objetivo de este trabajo es imitar técnicas de ecualización conocidas con naturaleza lineal, como los ecualizadores MSE, LMMSE y no linealies como MAP. Durante las pruebas realizaremos un seguimiento de diferentes tamaños de constelaciones y tasas de error de bit en una variedad de situaciones de interferencia y ruido. Esperamos que la tasa de error de bits disminuya hasta, o funcione ligeramente mejor que, el modelo de oro, que son los ecualizadores mencionados previamente.

Los últimos enfoques en investigación literaria demuestran la existencia de diversas estrategias que se aproximan a lograr una buena ecualización, aunque no registran todas las etapas de la ecualización basándose en métodos tradicionales. Por otro lado, se presentará una estrategia que incluso aborde condiciones más complicadas del canal, tales como canales de línea de vista o la falta de ella. Finalmente, pero no menos importante, se busca promover una mayor investigación en este campo prácticamente inexplorado por algunos equipos de comunicación.

Abstract

Due to their primary uses in control, road safety, autonomous driving, remote monitoring of vehicles, drone flying, and 6G needs, the development of high-mobility wireless communication networks is cutting-edge technology. The RF broadcasts, which are used by telecommunications systems to transfer encoded information over the amplitude and phase of a carrier signal, are normally called IQ data. However, this data is distorted as it travels through a high mobility channel. High mobility channels are doubly dispersive, which means that the signal properties are mixed with some temporal frequency shifting and spreading. There are some traditional algorithms that may require complex iterative equalizers that can be slow in terms of execution time. However, there are also simpler equalizers that may not be as precise. In this work, we will investigate the use of neural network-based solutions and consider how we should balance time complexity and accuracy to meet this challenge.

We can categorically assert that conventional algorithms continue to work well in the context of contemporary development. However, recent advances in neural networks have simplified problems that were previously intractable, such as deciphering DNA sequences or creating images from text. In short, they successfully tackle non-linear issues. The objective of this work is to mimic well-known equalization techniques with linear properties such as MSE and LMMSE, as well as non-linear techniques like MAP equalizers. We will evaluate how well the equalization performs under various noise conditions and with various bit-size encodings. We anticipate that the bit error rate will drop to, or perform slightly better than, the golden model, which are the aforementioned equalizers.

According to literature studies, there are a few cutting-edge methodologies that approach this goal but fall short of fully documenting all of the steps of equalization. This article presents a method for dealing with more complex circumstances, such as line of sight or a lack of it. Lastly, we aim to encourage further research into this largely untapped area by some communication teams.

Contents

1 Introduction

This chapter's main goal is to provide an overview of the technical facts and some of the current issues that prompted this study.

1.1 Document organization

The structure of this document is as follows:

- **Chapter 1 Theoretical Framework:** Fundamental ideas and concepts behind the current project. Outlines the core concepts necessary to comprehend this study
- **Chapter 2 State of Art:** Examines the existing literature on classical equalization methods and compares them to deep learning models.
- **Chapter 3 Present work:** The present work includes the Python implementation of various models, to train and test different experiments, as well as an overview of the software architecture that makes this possible. The focus is on explainable deep learning models of Quadrature Amplitude Modulation (QAM) equalization based on neural networks, including the design of the proposed models. Additionally, the description, characteristics, and comparison with related work will also be discussed
- **Chapter 4 Results and Analysis:** Explains the conception and execution of the idea, the experiments carried out, and the analysis of the outcomes. Here, we compare the BER (Bit Error Rate) graphs of our studied models.
- **Chapter 5 Conclusion and Outlook:** The conclusions and potential future directions of this study are determined.

1.2 Background

It takes a lot of expertise in the field of communications to model different types of channels, account for various channel flaws, and create the best signaling and detecting systems that guarantee a reliable data flow. The design of transmit signals in communications enables simple analytical techniques for symbol detection for a range of channel and system models, including multipath, doppler spread, and white Gaussian noise (AWGN) over constellation symbol detection. [1]

One of the guiding principles of communication system design is the division of signal processing into a series of distinct blocks, each of which performs a clearly defined and isolated functions such as source or channel coding, modulation, channel estimation, and equalization. This strategy has made it possible for us to have the effective, adaptable, and controlled systems we have today, but it is not certain whether individually tuned processing blocks will yield the best end-to-end performance [20]. During implementation, we

will also consider the abstraction of the other communication blocks and assume that the channel has already been estimated.

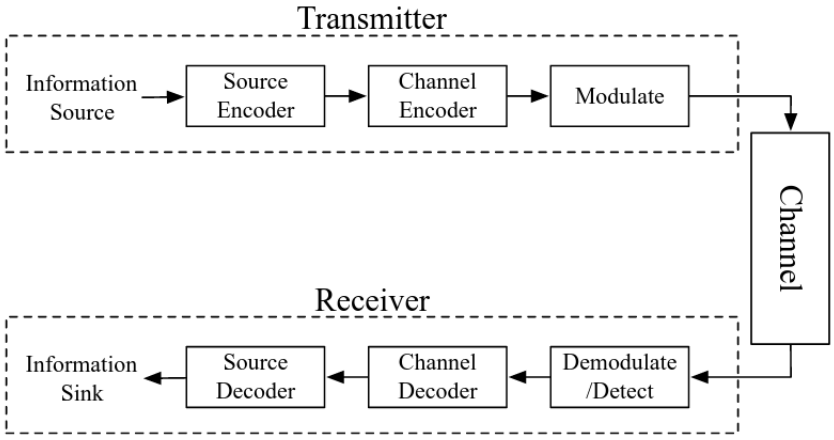

Figure 1: Typical communication system block diagram [6]

On the other hand, the design philosophy guiding the creation of the 6G architecture will be "AI native.", allowing the network to become intelligent, capable of solving individual stages of the channel, and able to self-learn and self-adapt in response to changing network dynamics [13]. It has been demonstrated that NNs are capable of approximating any function [11], and current research has demonstrated an astounding aptitude for algorithmic learning [25]. Due to the challenge of defining real-world images or language with strict mathematical models, deep learning (DL) excels in fields like computer vision and natural language processing. For example, while it is now simple to develop DL algorithms that learn to complete this task with accuracy greater than that of humans [9].

Despite this, we assume that the DL applications we evaluate in this thesis are a useful and insightful way to fundamentally rethink the problem of communications network design, and they show promise for performance improvements in complex communications scenarios that are difficult to describe with tractable mathematical models. Finally, we will look for the equalization stage, which, if it is seen in the diagram above, fits on the channel decoder.

1.3 OFDM

Orthogonal Frequency Division Multiplexing (OFDM) is a digital communication technique that divides the available bandwidth into a large number of narrowband subcarriers, and transmits data by modulating the subcarriers with symbols. OFDM is widely used in wireless and wired communication systems, such as Wi-Fi, LTE, and DOCSIS.

The basic blocks of an OFDM system are:

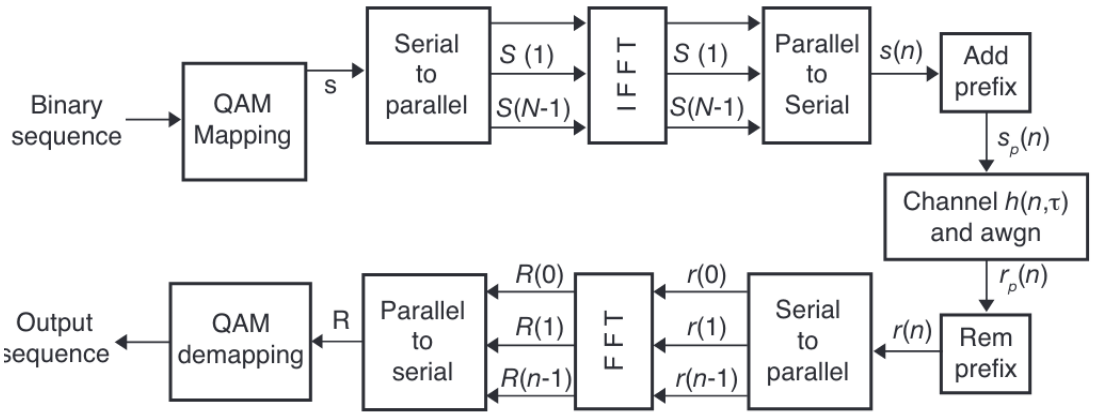

Figure 2: OFDM basic diagram [18]

- **Binary sequence**: This block generates the data to be transmitted. The data is usually a sequence of bits.
- **QAM mapping**: Bits are grouped into blocks called symbols usally named as alphabet $\mathbb{A}$. We have a set of bits $2^{\mathbb{A}}$ grouped inside the alphabet.
- **IFFT block**: This block applies an Inverse Fast Fourier Transform (IFFT) to the modulated symbols, dividing them into a set of subcarriers.
- **Cyclic prefix**: Helps OFDM symbols to reduce inter-symbol interference.

At regular intervals, the transmitter inserts recognizable symbols known as "pilots" into the transmitted signal. These pilots are selected to have a known value and to be separated from one another by a given number of symbols. By comparing the known transmission symbols to the received symbols, the receiver can then utilize the pilots to estimate the channel. This may lower the system's data throughput. Subsequently, the equalizer performs channel equalization using the coefficients obtained by the estimator, to reduce the distortions caused by the communication channel in the received data.

Since we have been provided with a dataset that contains previously estimated channels, we will not prioritize the channel estimation component of this project. Rather, in order to minimize errors and impairments at the receiver and to transmit the symbols as accurately as possible, we will focus on canceling the effects of the received, but well-known, channel during the communication process.

We are entering a new research area as we attempt to generalize the pseudo inverse of the channel to cancel its effects. It is well known that the matrix inverse is not a continuous function, as some matrices may not have an inverse or may be singular. However, we are using a subset of invertible matrices to mitigate this issue, and also considering realistic physical phenomena. Additionally, we must grapple with the numerical instability of the matrix inverse, as well as the use of complex numbers in a neural network.

1.3.1 Advantages and Disadvantages in OFDM

The advantages offered by OFDM systems in broadband systems are as follows [29]:

- **High spectral efficiency**: OFDM can transmit a large amount of data over a wide frequency band by dividing the available bandwidth into multiple narrowband subcarriers.
- **Robustness to channel impairments**: OFDM is less sensitive to frequency-selective fading and interference than other multiplexing techniques, making it well-suited for use in wireless communication systems.
- **Ease of implementation**: OFDM can be implemented using simple digital signal processing techniques, making it relatively easy to design and implement.

Some disadvantages of OFDM include:

- **High peak-to-average power ratio**: OFDM signals have a high peak-to-average power ratio, which can cause problems in power amplifier systems and limit the range of the transmitted signal.
- **Sensitivity to timing errors:** OFDM is sensitive to timing errors, which can cause inter-symbol interference and reduce the performance of the system.
- **Sensitivity to Doppler spread:** Doppler spread causes interference between the subcarriers.

1.4 QAM and IQ data

QAM (Quadrature Amplitude Modulation) is a type of digital modulation that encodes data onto a carrier signal by modulating the amplitude and phase of the signal. It is commonly used in digital communication systems to transmit digital data over analog channels. IQ data refers to the in-phase and quadrature components of a complex-valued signal. In digital communication systems, the IQ data is typically used to represent the amplitude and phase of the modulated carrier signal. It's ussualy represented with complex numbers in an alphabet $\mathbb{A} \in \mathbb{C}^n$.

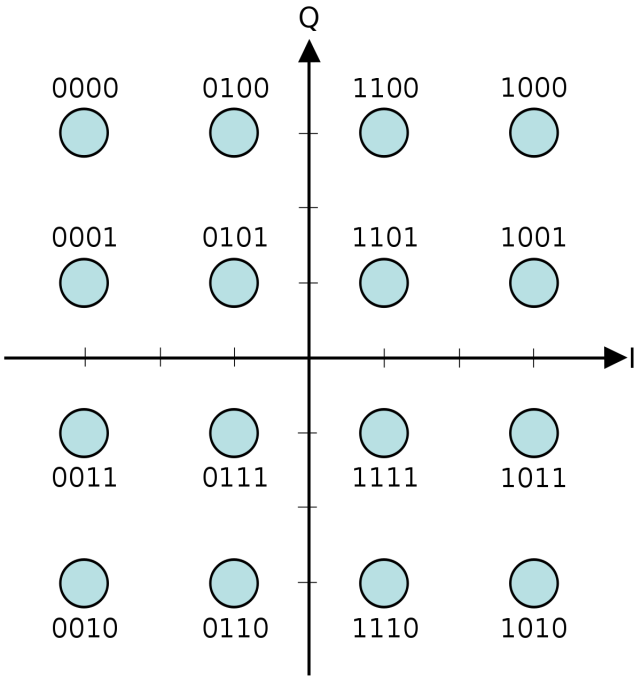

Figure 3: 16-QAM constellation [28]

In the field of communication systems, a QAM constellation refers to a graphical representation of the symbol points in an alphabet on the complex plane. Each point represents a specific combination of the in-phase and quadrature components of the modulated signal. The number of points in the constellation is determined by the number of possible combinations of in-phase and quadrature values, which is in turn determined by the number of bits per symbol used in the QAM modulation scheme. For instance, in a 16-QAM constellation, there are 16 symbol points arranged in a square grid, with each point corresponding to 4 bits of data. The distance between points in the constellation serves as an indicator of the signal-to-noise ratio required for reliable transmission of the data. It is common for QAM constellations to utilize gray code for assigning the symbol points in a manner that minimizes the error rate. However, this is not the only method that can be employed for this purpose

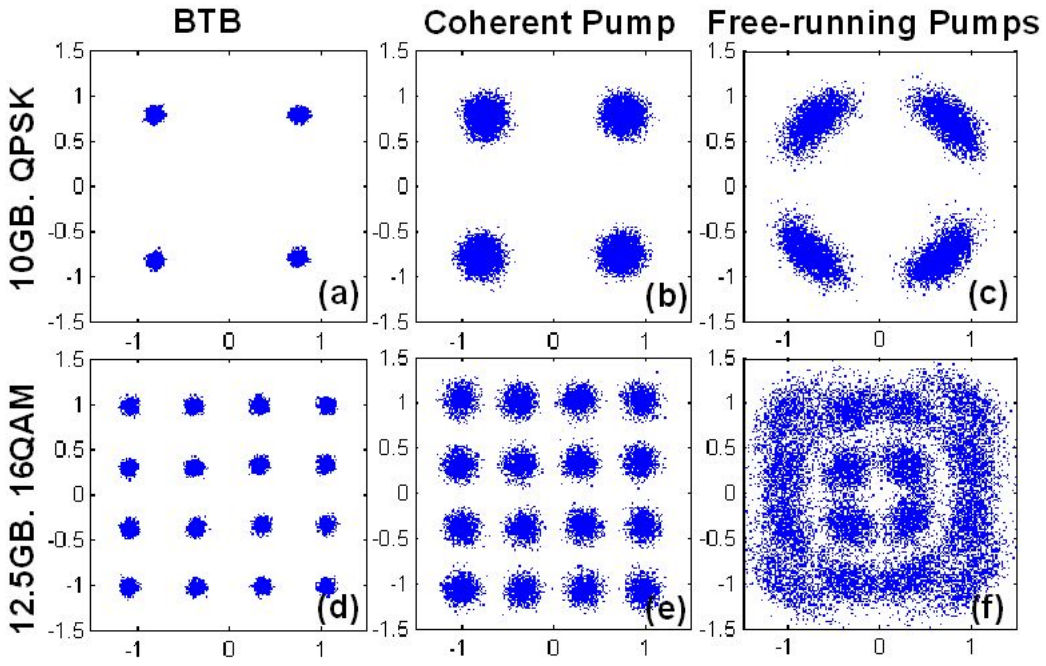

Figure 4: QPSK and 16 QAM with noise and phase displacement [14]

1.4.1 SNR

SNR stands for Signal-to-Noise Ratio which contrasts the strength of the signal with the strength of the noise. The most common way to measure it is in decibels (dB). In general, higher numbers indicate a better specification because there is a greater ratio of useful information (the signal) to unwanted data (the noise). The following equation displays the necessary power spectral density of the noise. [20]

$$N_0 = \frac{P_{signal}}{P_{noise}} \tag{1}$$

$$P_{signal} = \sum |s|^2 \tag{2}$$

For realistic simulations, noise is typically an AWGN, which is useful. In order to simulate bit error rates, AWGN can generate appropriate power levels. Typically, noise power is determined by its variance σ^2, which in simulations is used as follows.

$$\sigma^2 = 10^{\frac{SNR_{dB}}{10}} \tag{3}$$

Misunderstandings have arisen regarding the signs assigned to the positive and negative exponents in the equation. While there are conflicting explanations in various sources,

to clear up any confusion, it's important to consider the visual representation of the equation

$$N_0 = \frac{P_{signal}}{P_{noise}} = \frac{\sum |s|^2}{10^{\frac{SNR_{dB}}{10}}} = \sum |s|^2 \times 10^{-\frac{SNR_{dB}}{10}} \tag{4}$$

The higher the SNR, the better the quality of the signal, since the noise is relatively weaker. In practice, the SNR is usually calculated for a specific frequency band or for a specific signal-processing method.

The necessary noise variance (noise power) for producing Gaussian random noise, assuming complex IQ plane for all digital modulations, is given by

$$n = \sqrt{(N_0/2)} * (randn(size(s)) + j * randn(size(s))) \tag{5}$$

with the recived signal

$$r = s + n \tag{6}$$

1.5 Channel

1.5.1 Dispersion and doubly dispersion

Dispersion, in the context of radio communication systems, refers to the spreading out of a signal over a range of frequencies or wavelengths as it travels through a medium, such as air. This phenomenon can lead to distortion of the signal and negatively impact the performance of the communication system. There are various factors that can contribute to dispersion, including the distance the signal travels, the presence of obstacles or reflections, and the frequency of the signal. Techniques such as equalization, adaptive modulation, and error correction codes can be used to mitigate the effects of dispersion. A doubly dispersive channel is a type of communication channel that exhibits dispersion in two dimensions, such as time and frequency. This means that the signals transmitted through the channel are spread out in both time and another aspect, such as frequency or spatial dimensions. [21]

1.5.2 Multipath fading and doppler shift

A multipath fading channel is a type of wireless communication channel that experiences fading of the transmitted signal due to multiple paths of the signal between the transmitter and receiver. This can occur when the signal reflects off of obstacles such as buildings

or terrain, or when it is refracted by the atmosphere. Multiple paths of the transmitted signal can cause constructive and destructive interference at the receiver, resulting in rapid fluctuations in the received signal strength. This can cause the signal to fade in and out, which can affect the quality and reliability of the communication. [3]. We begin with the ray-tracing technique and make advantage of the physical geometry of the propagation environment to create a deterministic model of the wireless channel. The delay of a signal refers to the time it takes for the signal to travel from the transmitter to the receiver, and the Doppler shift of a signal refers to the frequency shift of the signal due to the relative motion between the transmitter and receiver.

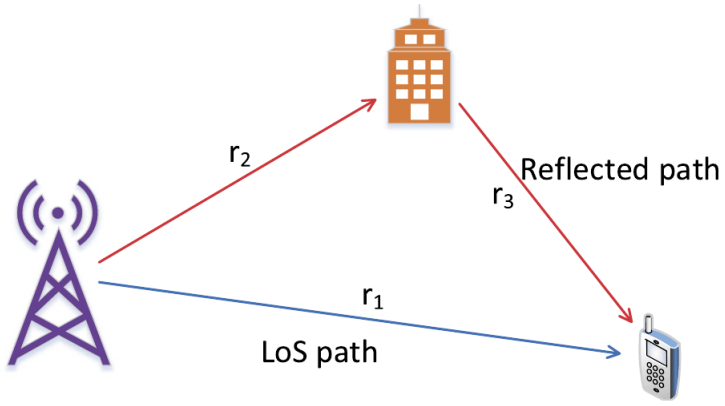

Figure 5: Delay multipath [10]

$$\mathbf{r(t) = g_1 s(t - \tau_1) + g_2 s(t - \tau_2)} \tag{7}$$

- $\mathbf{r(t)}$ recieved signal
- $\mathbf{g_n}$ baseband equivalent complex gain(attenuation)
- $\boldsymbol{\tau_1} = \frac{r_1}{c}$ where c is the speed of light
- $\boldsymbol{\tau_2} = \frac{(r_2 + r_3)}{c}$ delay in reflected path
- $\boldsymbol{\tau_2 - \tau_1}$ delay spread

In wireless communication, the Doppler shift can cause changes in the frequency of a signal as it travels from the transmitter to the receiver. This can happen when the transmitter or receiver (or both) are moving relative to each other, causing the frequency of the signal to shift. The Doppler shift can affect the performance of a wireless communication system by causing changes in the signal-to-noise ratio and the signal-to-interference ratio, which can degrade the quality of the signal and make it more difficult to detect and decode.[15]

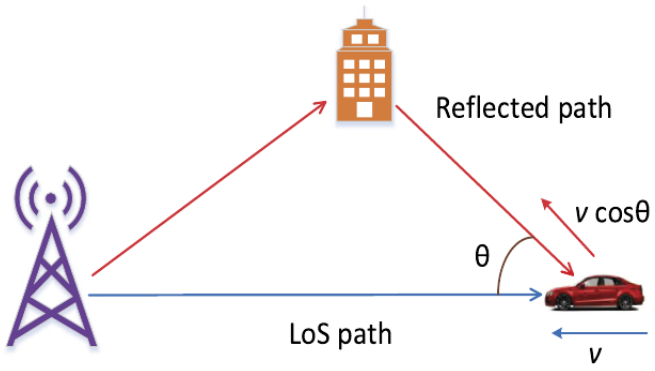

Figure 6: Doppler shifts due to the different angles of arrival [10]

$$\mathbf{r(t)} = \mathbf{g_1 e^{j2\pi\nu_1(t-\tau_1)} s(t-\tau_1) + g_2 e^{j2\pi\nu_2(t-\tau_2)} s(t-\tau_2)} \tag{8}$$

- $\boldsymbol{\nu_1} = \frac{v}{c}\boldsymbol{f_c}$ LOS doppler shift
- $\boldsymbol{\nu_1} = \frac{\boldsymbol{v*cos(\theta)}}{\boldsymbol{c}}\boldsymbol{f_c}$ doppler shift in reflected path

We can generalize for time dependent function the gain as follows:

$$\mathbf{g(\tau_i, t) = g_i e^{j2\pi\nu_i(t-\tau_i)}} \tag{9}$$

Therefore impulse time-frequency response of channel at fixed time t, can be obtained by taking fourier transform along the delay dimension of $\boldsymbol{g(\tau, t)}$

$$\boldsymbol{H(f,t) = \int g(\tau,t) e^{-j2\pi f\tau} d\tau} \tag{10}$$

A time-frequency channel is a type of communication channel that is characterized by time-varying frequency-selective fading. This means that the channel experiences changes in its frequency response over time, resulting in variations in the amplitude and phase of the signals transmitted through it. Because a channel is assumed to have a slow time-varying function of t, we refer to this phenomenon as having a wide sense of being stationary. [21]

1.5.3 Wide sense stationary

A wide sense stationary (WSS) process is a stochastic process in which the mean and the autocorrelation function of the process do not change over time. As a result, the process' statistical characteristics, such as the mean, variance, and autocorrelation, remain consistent across time. In signal processing and telecommunications, WSS processes are

frequently used to simulate signals that are stationary over a period of time. A generalization of strictly stationary processes, or processes in which the mean and auto-correlation function are constant over time, are WSS processes. [19]

A wide sense stationary (WSS) process can be described by the following equations:

1. The mean of the process is constant over time, and can be represented by the equation:

$$\boldsymbol{E}[\boldsymbol{X}(\boldsymbol{t})] = \boldsymbol{\mu} \tag{11}$$

   Where $\boldsymbol{E}[\boldsymbol{X}(\boldsymbol{t})]$ is the expected value of the process at time t, and $\boldsymbol{\mu}$ is the constant mean of the process.

2. The autocovariance of the process, which is a measure of the correlation between two points in time, is also constant over time, and can be represented by the equation:

$$\boldsymbol{R_X}(\boldsymbol{t1}, \boldsymbol{t2}) = \boldsymbol{E}[(\boldsymbol{X}(\boldsymbol{t_1}) - \boldsymbol{\mu})(\boldsymbol{X}(\boldsymbol{t_2}) - \boldsymbol{\mu})] = \boldsymbol{R}(\boldsymbol{t_1} - \boldsymbol{t_2}) \tag{12}$$

   Where$\boldsymbol{R_X}(\boldsymbol{t_1}, \boldsymbol{t_2})$is the autocovariance of the process at times t1 and t2, and $\boldsymbol{R}(\boldsymbol{t_1} - \boldsymbol{t_2})$ is the autocovariance function of the process, which is a function of the time difference between t1 and t2

3. The autocorrelation function of the process, which is a measure of the correlation between two points in time, is also constant over time, and can be represented by the equation

$$\boldsymbol{r_X}(\boldsymbol{t1}, \boldsymbol{t2}) = \frac{\boldsymbol{R_X}(\boldsymbol{t_1}, \boldsymbol{t_2})}{\sqrt{\boldsymbol{R_X}(\boldsymbol{t_1}, \boldsymbol{t_1})\boldsymbol{R_X}(\boldsymbol{t_2}, \boldsymbol{t_2})}} = \boldsymbol{r}(\boldsymbol{t_1} - \boldsymbol{t_2}) \tag{13}$$

   Where $\mathbf{r_X}(\mathbf{t_1}, \mathbf{t_2})$ is the autocorrelation function of the process at times t1 and t2, and $\boldsymbol{r}(\boldsymbol{t_1} - \boldsymbol{t_2})$ is the autocorrelation function of the process, which is a function of the time difference between t1 and t2.

1.6 Equalization

A telecom channel equalizer is a device or algorithm used in telecommunications systems to compensate for distortion or other impairments in a communication channel. The equalizer uses signal processing techniques to estimate the characteristics of the channel, such as the impulse response or the frequency response, and then applies a correction to the

transmitted signal to counteract the effects of the channel on the received signal. This can improve the performance of the communication system by reducing errors and increasing the data rate or signal-to-noise ratio. Bluetooth, WiFi, IOT, drones, V2V, wireless broadband, and satellite communications are just a few of the everyday applications they can be used for. [7]

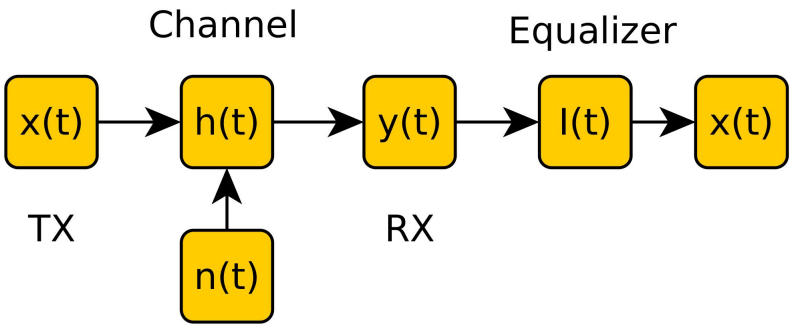

Figure 7: Basic Equalizer

Our goal is to develop an equalizer based on deep learning techniques and research how it performs in terms of timing and memory complexity. We take as a reference the classical methods that manage a good bit error rate, and we will take them as a golden model of accuracy.

1.6.1 MSE

MSE equalization, or minimum mean square error equalization, is a technique used in digital communication systems to improve the performance of a channel by reducing the mean square error (MSE) between the transmitted and received signals. This equalization is a widely used technique in digital communication systems, as it can provide significant improvements in the performance of the channel by reducing the effects of noise, interference, and other impairments. It is often used in combination with other techniques, such as error correction and modulation, to further improve the overall performance of the communication system.[20]

$$\boldsymbol{MSE} = \boldsymbol{E}[(\boldsymbol{x} - \boldsymbol{y})^{\boldsymbol{2}}] \tag{14}$$

- x is the transmitted signal
- y is the received signal.

1.6.2 MMSE

Since MMSE equalization is a linear equalization technique, the transmitted signal is first adjusted using a linear filter before being sent across the channel. By choosing the proper filter coefficients for the linear filter, the purpose of MMSE equalization is to reduce the MSE between the sent and received signals. The MMSE criteria, which stipulates that the filter coefficients should be set to minimize the MSE between the sent and received signals, may be used to compute the filter coefficients. The equation below can be used to do this:

$$\boldsymbol{H} = \frac{\boldsymbol{E}[\boldsymbol{xy}]}{\boldsymbol{E}[\boldsymbol{x^2}]} \tag{15}$$

- $\boldsymbol{H}$ is the filter coefficient.
- x is the transmitted signal.
- y is the received signal.

It is possible to modify the sent signal before it is transmitted across the channel by using the weights determined values in the linear filter. The signal is modified and provided by:

$$\hat{\boldsymbol{x}} = \boldsymbol{H}\boldsymbol{x} \tag{16}$$

- $\hat{x}$ is the adjusted signal and x is the original transmitted signal.

1.6.3 LS

Least squares (LS) equalization is a linear equalization method that aims to minimize the mean squared error (MSE) between the estimated and the transmitted symbols. The noise component of x is disregarded by the LS (Least-Squares) equalizer, which treats it as a deterministic variable. A linear filter $\boldsymbol{W}$ that minimize the mean square error between $\hat{\boldsymbol{x}}$ and $\boldsymbol{H}\boldsymbol{x}$.

The argmin function is a mathematical function that returns the indices of the minimum values of a given array or matrix. It is often used in optimization problems, where it is used to find the values of the variables that minimize some objective function.[17]

$$\boldsymbol{W_{Ls} = argmin||\hat{x} - Hx||^2} \tag{17}$$

With the solution in the Moore-Penrose pseudoinverse $\boldsymbol{H^+}$.The Moore-Penrose pseudoinverse is often used to solve linear least squares problems, which involve finding the values of variables that minimize the sum of the squares of the residuals (the differences between the observed values and the values predicted by the model). It can also be used to compute a "best fit" solution for systems of linear equations that do not have a unique solution. [26]

$$\boldsymbol{H^+ = (H^H H)^{-1} H^H} \tag{18}$$

- $\boldsymbol{H}$ Channel matrix
- $\boldsymbol{H^+}$Moore-Penrose pseudoinverse
- $\boldsymbol{H^H}$ Hermiatian transpose matrix. Complex square matrix that is equal to its own conjugate transpose

Its resistance to noise makes it appealing in a variety of communication systems, however because the noise component is ignored, it cannot operate satisfactorily with low SNR (signal to noise ratio).

1.6.4 LMMSE

A form of linear filter called the Linear Minimum Mean Squared Error (LLMSE) Equalizer seeks to reduce the mean squared error (MSE) between the actual signal $\boldsymbol{x}$ and an estimation of the signal $\boldsymbol{\hat{x}}$. Incorporating second-order statistics of the data and the noise and treating the noise as a random variable allows it to achieve this. Compared to the Least Squares (LS) algorithm, this can perform better when there is low signal-to-noise ratio (SNR). To put it another way, the LLMSE equalization is a method that can be applied to restore precision to a signal that has been distorted by noise, especially when the noise level is high. When there is a low signal-to-noise ratio (SNR) and a significant quantity of noise in the signal, it performs exceptionally well.

$$\boldsymbol{W_{LMMSE} = argminE||x - \hat{x}||^2} \tag{19}$$

This equation consider AWGN (Additive White Gaussian Noise) with variance $\boldsymbol{\sigma^2}$. It is called "white" because it has a flat power spectral density, meaning that it has equal power at all frequencies. It is called "additive" because it can be added to a signal without changing its distribution.

$$\boldsymbol{W_{LMMSE} = (H^H H + \sigma^2 I)^{-1} H^H} \tag{20}$$

- $\boldsymbol{\sigma^2}$ variance of AWGN

1.6.5 Pros and Cons

As we can see in the previous methods, the calculation of the equalization matrices is mainly based on finding the pseudo-inverse of certain matrix factors. However, directly solving for the inverse of the matrix can be computationally complex $O(N^3)$ and, in addition, numerically unstable. This occurs if the matrix has a very small determinant, in which case the true solution may be subject to large perturbations. This will lead to a very complicated circuit architecture for numerical calculation.

1.7 Neuronal networks

Neural networks are a type of artificial intelligence system designed to mimic the functioning of the human brain. They consist of interconnected nodes, or "neurons," which are capable of processing information and making decisions based on that information. These networks are usually organized into layers, with each layer containing a different number of neurons. The input layer receives input from the external environment, while the output layer produces the final result or decision based on that input. The layers in between the input and output layers are called hidden layers, and they perform various intermediate calculations and processing tasks. Neural networks are trained using large amounts of data, allowing them to learn and make predictions or decisions based on that data. In this study case, the data consists of realistic channel realizations.

1.7.1 Linear Layer

A linear layer in a neural network is a type of layer that applies a linear transformation to the input data. This transformation can be represented by a **matrix** of weights, denoted as $\boldsymbol{W_n}$,and a biases **vector**, denoted as $\boldsymbol{b_n}$, which are learned during the training process. The subscript $\boldsymbol{n}$ refers to the nth layer. The output of a linear layer is calculated by performing a matrix and vector product between the input data $\boldsymbol{X_{n-1}}$ and the weights $\boldsymbol{W_n}$ as well as adding the biases.

$$\boldsymbol{X_n = W_n X_{n-1} + b_n} \tag{21}$$

- $\boldsymbol{W_n}$weight matrix at layer n.
- $\boldsymbol{b_n}$ bias at layer n
- $\boldsymbol{X_{n-1}}$Input or last layer data
- $\boldsymbol{X_n}$ Output data

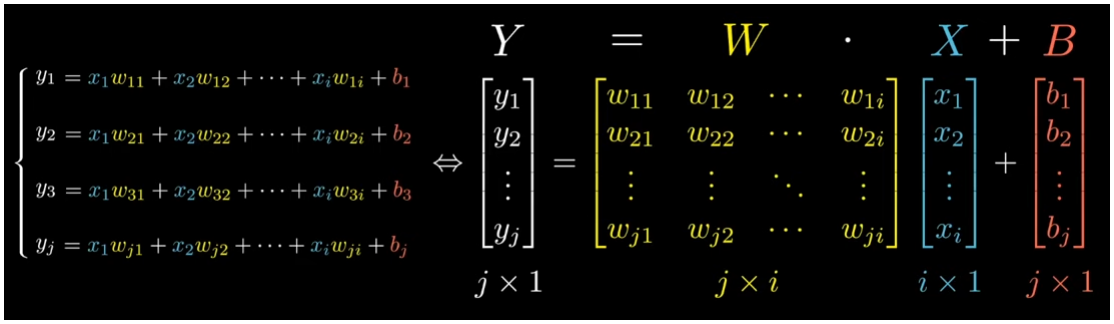

Figure 8: Output as Y and input as X. Vector matrix representation of system. Desing done with manim

1.7.2 Backpropagation

The goal of the layer is to optimize the weights parameters so that they fit a target referred to as the ground truth. The error, denoted by E, is calculated as the difference between the predicted output ($\hat{X}$) and the actual target output (X_n). To achieve this, we will utilize the backpropagation algorithm, which is a common method in the field of artificial neural networks for training the network by adjusting the weights between neurons in the network.

$$\boldsymbol{Err = \hat{X} - X_n} \tag{22}$$

This method analyzes the error rate in relation to the weights and inputs. As we adjust our trainable parameters, {$\boldsymbol{W_n}$, $\boldsymbol{b_n}$}, the error will change accordingly. The goal is to minimize the error, or to find a point where the error gradient is zero.[8]

$$\boldsymbol{\nabla W = \frac{\partial E}{X_{n+1}} \times X_{n-1}^T} \tag{23}$$

$$\boldsymbol{\nabla X_{n-1} = W_n^T \times \frac{\partial E}{X_{n+1}}} \tag{24}$$

- $\boldsymbol{\nabla W}$Gradient of weights.The gradient is a multi-variable generalization of the derivative.

1.7.3 Learning rate

During the process of updating the weights, the learning rate is a hyperparameter that determines the size of the update step applied during the training of a neural network.. It is a scalar value that is used to multiply the gradient of the loss function with respect to the weights of the network during gradient descent. The learning rate determines how fast the weights of the network are updated during training. A smaller learning rate results in slower updates, while a larger learning rate results in faster updates. The learning rate is typically set manually, and finding the optimal learning rate for a given problem is an important aspect of training a neural network. [22]

$$\boldsymbol{W_n} = \boldsymbol{W_n} - \gamma \boldsymbol{\nabla W} \tag{25}$$

$$\boldsymbol{b_n} = \boldsymbol{b_n} - \gamma \frac{\boldsymbol{\partial E}}{\boldsymbol{X_n}} \tag{26}$$

If the learning rate is set too low, the optimization process may become stuck in a local minimum or local maximum. A local minimum is a point in the optimization landscape where the cost function has a lower value than the surrounding points, but is not the global minimum. A local maximum is a point where the cost function has a higher value than the surrounding points, but is not the global maximum. This can lead to suboptimal performance or even failure of the optimization process. On the other hand, if the learning rate is set too high, the optimization process may oscillate or diverge, also leading to suboptimal performance. It is important to choose an appropriate learning rate for the optimization process in order to avoid these problems.

1.7.4 Basic code implementation

Although it may seem difficult to implement the code, it is actually quite simple as it involves encapsulating the results of the previous layer's calculations.

```
import numpy as np
from layer import Layer

class Linear(Layer):
    def __init__(self, input_size, output_size):
        self.weights = np.random.randn(output_size, input_size)
        self.bias = np.random.randn(output_size, 1)

    def forward(self, input):
        self.input = input
        return  self.weights@self.input + self.bias

    def backward(self, output_gradient, learning_rate):
        weights_gradient = output_gradient @ self.input.T
        input_gradient   = self.weights.T  @ output_gradient
        self.weights    -= learning_rate * weights_gradient
        self.bias       -= learning_rate * output_gradient
        return input_gradient
```

And a basic implmentation of the Linear Class.

```
from Linear import Linear
INPUT_SIZE    = 784
MIDDLE_SIZE   = 300
OUTPUT_SIZE   = 10

H1 = Linear(INPUT_SIZE,MIDDLE_SIZE)
H2 = Linear(MIDDLE_SIZE,OUTPUT_SIZE)

def forward(X):
    A1  = H1.forward(X)
    A2  = H2.forward(A2)
    return A2

#gamma learning rate
def backward(ERR,gamma):
    #internamente de estas funciones
    #se van corrigiendo los pesos
    B2 = H2.backward(ERR,gamma)
    B1 = H1.backward(B2,gamma)
```

1.7.5 Activation functions

Activation functions are used in neural networks to introduce non-linearity into the network. This is important because many real-world problems are non-linear in nature and a neural network with only linear functions would not be able to model such problems accurately. Activation functions allow the network to learn more complex patterns in the data and improve the accuracy of the network. They also help to prevent the network from becoming stuck in a local minimum or plateau during training.

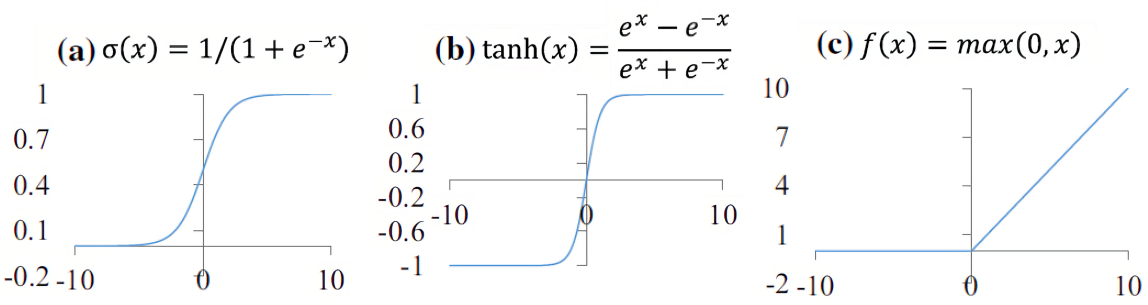

Figure 9: Activation function. (a) Sigmoid, (b) tanh, (c) ReLU. [27]

The most popular activation functions in neural networks are the sigmoid function, the hyperbolic tangent (tanh) function, and the rectified linear unit (ReLU) function. The sigmoid function maps any input value to a range between 0 and 1, while the tanh function maps input values to the range between -1 and 1. The ReLU function is a linear function that maps all negative input values to 0 and all positive input values to their original value.[24]

However, there are "hardened" versions of activation functions that can be evaluated

computationally faster yet produce similar results. Examples of these include hardtanh and hardsigmoid. The Hardtanh function is defined as:

$$\boldsymbol{hardtanh}(x) = \begin{cases} -1, & \text{if } x < -1 \\ x, & \text{if } -1 \leq x \leq 1 \\ 1, & \text{if } x > 1 \end{cases} \tag{27}$$

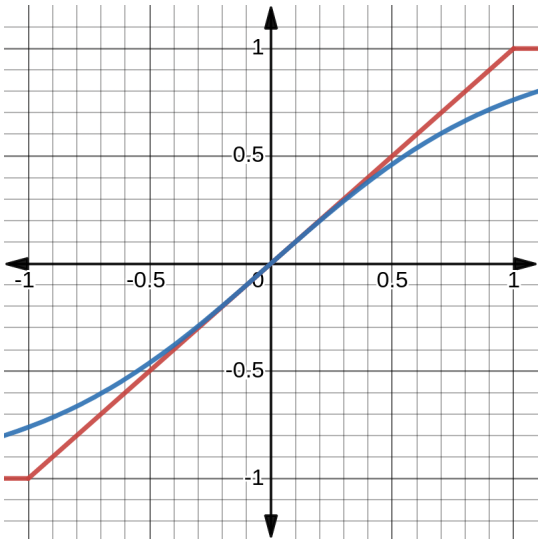

Figure 10: Hardtanh(red) and Tanh(blue)

1.8 Effective Techniques for Improving Model Generalization

In the field of machine learning, it is important to ensure that the models we build are able to accurately and effectively make predictions on new data. However, it is common for models to suffer from issues such as overfitting or poor generalization to new data. In this section, we will explore three techniques that can be used to improve the performance of machine learning models: **regularization**, **normalization**, and **standardization**. By properly applying these techniques, we can mitigate the risks of overfitting and improve the ability of our models to generalize to new data. [5, 2, 23]

Our data set, which includes doubly dispersive channels of a complex nature, requires extra attention. We will use distorted vectors or inverse matrices, which can result in numerical instability, as our ground truth. These extra precautions will help to guarantee the process' success. Ignoring these recommendations may result in unsatisfactory outcomes, as the neural network may not generalize as well and may perform poorly. [4]

1.8.1 Regularization

Regularization is a technique used in machine learning to prevent overfitting. Overfitting occurs when a model is overly complex and captures the noise in the training data, rather than the underlying relationships. This results in poor generalization to new data. Regularization works by adding a penalty term to the objective function that the model is trying to minimize. This penalty term discourages the model from learning relationships that are too complex, and encourages it to learn simpler relationships that generalize better. There are several methods for regularization, including [8]:

1. L1 regularization: This method adds a penalty term to the cost function that is proportional to the absolute value of the weights.
2. L2 regularization: This method adds a penalty term to the cost function that is proportional to the square of the weights.
3. Dropout regularization: This method randomly sets a fraction of the weights in the model to zero during training, which helps to prevent overfitting by reducing the number of parameters in the model. Dropout is only applied during the training process, and all neurons are available during evaluation.

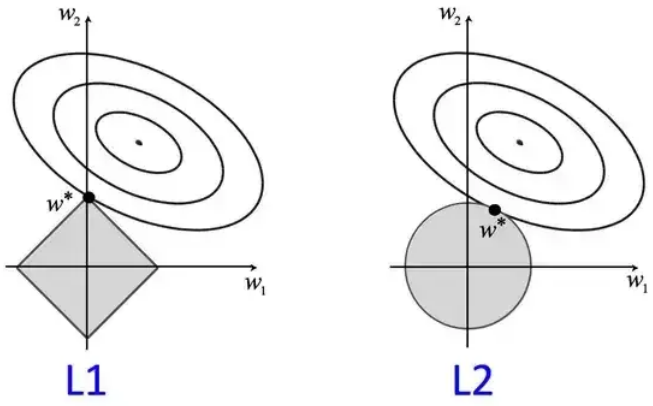

Figure 11: Level sets of the loss function and L1,L2 regularization [16]

To implement regularization, you can modify the cost function of the model to include the regularization term. For example, in L2 regularization, the cost function would be modified to include the sum of the squares of the weights, as shown in the following equation:

$$\boldsymbol{J(W)} = \frac{\boldsymbol{\lambda}}{\boldsymbol{2}}||\boldsymbol{w}||^{\boldsymbol{2}} = \frac{\boldsymbol{\lambda}}{\boldsymbol{2}}\sum_{\boldsymbol{j=1}}^{\boldsymbol{m}} \boldsymbol{w_j^2} \tag{28}$$

- $\boldsymbol{\lambda}$ regularization parameter
- $\boldsymbol{J(W)}$ Cost function

1.8.2 Normalization

Data is scaled using a process called normalization to give it a unit norm (or length). This is frequently done to improve the data's suitability for particular machine learning algorithms, such as those that use gradient descent or have a set range for acceptable input data. When comparing various features, normalization can also be used to scale down the data to a common scale. Data normalization methods include min-max normalization, mean normalization, and z-score normalization.

Normalization of complex numbers involves dividing a complex number by its magnitude (or absolute value) to obtain a complex number with a magnitude of 1. This is typically done to simplify calculations and make it easier to compare complex numbers. The normalized form of a complex number is often written as ẑ = z/|v|, where z is the original complex number and ẑ is the normalized form and |v| is the max magnitud of the entire vector of the complex numbers. Normalization of complex numbers is useful in many applications, including signal processing and control systems, where it is often necessary to compare complex numbers on an equal footing. [4]

$$\hat{\boldsymbol{z}} = \frac{\boldsymbol{z}}{|\boldsymbol{v}|} \tag{29}$$

- $|v|$ Maximum magnitud of the vector
- z Input value
- $\hat{z}$ Normalized value

1.8.3 Standarization

Standardization is a method used in machine learning to transform the values of a feature or set of features to a standard scale. The standard scale is typically defined as having a mean of 0 and a standard deviation of 1. Standardization is often used as a preprocessing step before training a model, as it can help to improve the performance and convergence of the model. Standardization can be useful when the features in the dataset have different scales or units, as it can help to bring them onto a common scale and make it easier for the model to learn from the data. Standardization can be applied to both real and complex-valued data.

$$\hat{\boldsymbol{z}} = \frac{\boldsymbol{z} - \boldsymbol{\mu}}{\boldsymbol{\sigma}} \tag{30}$$

- μ Mean of the data
- σ Standard deviation of the data
- z Input value
- $\hat{z}$ Standarized value

1.9 Convolutional Neuronal Networks

A convolutional neural network (CNN) is a type of deep learning neural network that is primarily used for image and video recognition. It is designed to process data that has a grid-like topology, such as an image. The network is composed of multiple layers, including **convolutional layers**, **activation layers**, **pooling layers** and **linear layers**.

The convolutional layers apply a set of filters to the input data, where each filter is a small matrix of weights. This ones are used to identify features in the data such as **edges**, **textures**, and **shapes**, specifically, we will be utilizing these layers to extract the relationship of intercarrier symbol interference (ISI) and to perform dimensionality reduction. [12][8]

- The **activation layers** or activation functions introduce non-linearity to the network, allowing it to learn complex representations of the input data.
- The **pooling layers** reduce the spatial dimensions of the data, which helps to reduce overfitting and computational cost.
  - Max pooling is used to pick the feature with the highest activation in a small region of the input feature map
  - Average pooling is used to reduce the spatial size of the input data by taking the average of the values of a small region of the input feature map.

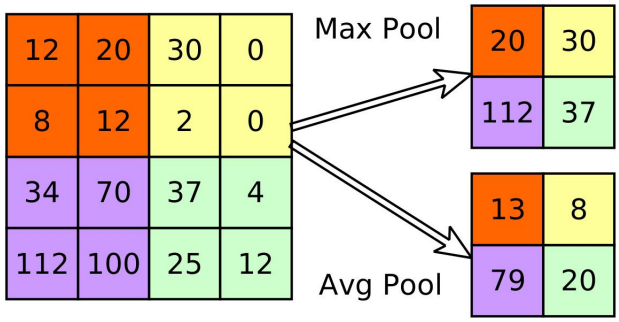

Figure 12: Average and max pooling

- **Linear layers** classify the features extracted by the convolutional layers into the desired output.

It should be noted that what is commonly referred to as 'convolution' in the context of convolutional neural networks (CNNs) is actually a cross-correlation operation, denoted with symbol $\star$ and convolution usually is used $*$. The term 'convolution' is used only for convention purposes. The basic concept behind cross-correlation is to take a small matrix, referred to as a kernel or filter, and slide it over the input data (such as an image or audio signal). At each position, the kernel is multiplied element-wise with the underlying data, and the results are summed to produce a single output value, referred to as a feature map.

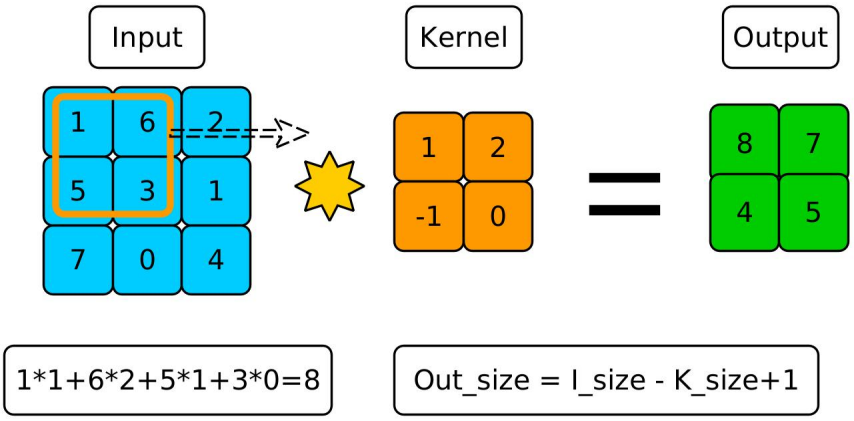

Figure 13: Basic cross-correlation operation

The output of the convolution operation is a feature map, where each element in the feature map is computed as follows:

$$\boldsymbol{Y}_{ij} = \sum \boldsymbol{K}_{ab} \circ \boldsymbol{I}_{i+a,j+b} \tag{31}$$

Where kernel(a,b) is the value of the filter at position (a,b), input(i+a,j+b) is the value of the input at position (i+a,j+b) and output(i,j) is the output value at position (i,j)

1.9.1 Channels

A channel refers to a specific feature or dimension of the input data. For example, in the case of image data, a channel can represent a color channel such as red, green, or blue. These channels are used to extract different features of the input image, and they are processed separately by the CNN. In our case study, we can use channels as a division between the real and imaginary parts, or for feature extraction of intercarrier symbol interference (ISI). We can have N channels as input and M channels as output, depending on how many features we want to deal with. In the image below, we show a case of 3 channel input and two channel output, also with a bias term.

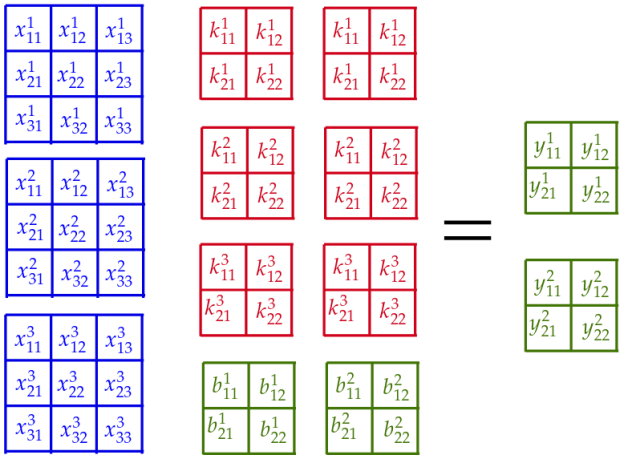

Figure 14: Convolutional Neural Network with 3-channel Input and 2-channel Output, including bias term

1.9.2 A Generalized View of Linear Layers through Convolutional Neural Networks

Let's take a more detailed look at the math, given the following terms.

- j input channels with$\boldsymbol{X}_j$ matrices
- d ouput channels $\boldsymbol{Y}_d$matrices and this ones an output size of $\boldsymbol{X}_j - \boldsymbol{K}_{ij}$
- $\boldsymbol{K}_{ij}$ kernels,where i maps to $\boldsymbol{Y}_d$ and j to $\boldsymbol{X}_j$
- $\bigstar$ cross-correlation

$$\boldsymbol{Y}_d = \boldsymbol{B}_d + \sum_{j=1}^{n} \boldsymbol{X}_j \bigstar \boldsymbol{K}_{ij} \tag{32}$$

We can think of the matrices as individual blocks and visualize them in the image below.

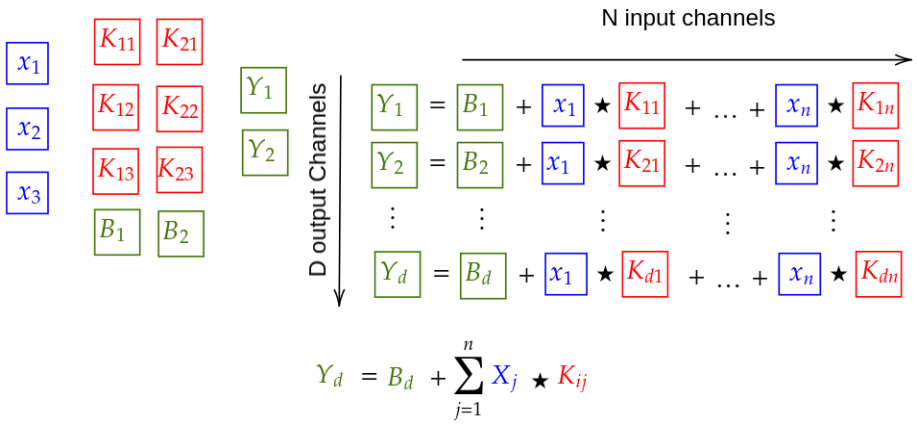

Figure 15: Y_d output given kernels

Finally, with the use of abstraction, we can further simplify the problem by representing it as a generalized version of tensors. The internal tensor is given by the sum of the cross-correlations between X channels and kernels. In a more general perspective, this can be viewed as the inner product of two tensors.

$$\boldsymbol{Y} = \boldsymbol{B} + \langle \boldsymbol{K}, \boldsymbol{X} \rangle \tag{33}$$

$$\begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_d \end{bmatrix} = \begin{bmatrix} B_1 \\ B_2 \\ \vdots \\ B_d \end{bmatrix} + \begin{bmatrix} K_{11} & K_{12} & \cdots & K_{1n} \\ K_{21} & K_{22} & \cdots & K_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ K_{d1} & K_{d2} & \cdots & K_{dn} \end{bmatrix} \dot{\times}\star \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_n \end{bmatrix}$$

$$Y = B + \langle K, X \rangle$$

Figure 16: Higher dimensionality abstract version

It's worth noting that when we consider a kernel of 1 dimension and an input of 1 channel, we can see that a dense layer is just a specific case of a 2D convolutional layer (Conv2D). Just as equation (15) and figure (8)

1.10 Motivation

1.11 Objectives

1.11.1 General Objective

We should analyze and test different techniques of neural networks applied to frequency

channel equalization, which is commonly used in OFDM. The NN models will be based on existing equalizers with well-known equations.

1.11.2 Particular Objective

Our objective is to equalize a double dispersive channel, also known as a frequency-time channel and intercarrier symbol interference (ISI) through the utilization of various neural network strategies. These strategies will focus on specific stages of the existing equalization process. We aim to leverage the non-linearity of neural networks and employ techniques such as dimensionality expansion and reduction. Our aim is to recover QAM and PSK constellation symbols that have been distorted by a channel with both line-of-sight (LOS) and non-line-of-sight (NLOS) conditions, plus Gaussian noise. Our ultimate goal is to achieve a bit error rate (BER) comparable to that of the "golden models" or to reduce the computational complexity of current methods.

2 State of Art

3 Present work

4 Results and Analysis

5 Conclusion and Outlook

5.1 Future Work

6 References

References

[1] Hamid Aghvami and Pheng-Ann Heng. *Channel Modeling for Wireless Communication Systems*. John Wiley Sons, 2005.

[2] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[3] Constantine A. Balanis. *Advanced Engineering Electromagnetics*. John Wiley amp Sons, 2012.

[4] Joshua Bassey, Lijun Qian, and Xianfang Li. A survey of complex-valued neural networks, 2021.

[5] Ekaba Bisong. *Regularization for Deep Learning*, pages 415–421. Apress, Berkeley, CA, 2019.

[6] Nariman Farsad, Weisi Guo, and Andrew Eckford. Tabletop molecular communication: Text messages through chemical signals. *PloS one*, 8:e82935, 12 2013.

[7] Andrea Goldsmith. *Wireless Communications*. Cambridge University Press, 2005.

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. pages 1026–1034, 2015.

[10] Yi Hong, Tharaj Thaj, and Emanuele Viterbo. *Delay-Doppler Communications: Principles and applications*. Academic Press, an imprint of Elsevier, 2022.

[11] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[13] Khaled B. Letaief, Wei Chen, Yuanming Shi, Jun Zhang, and Ying-Jun Angela Zhang. The roadmap to 6g: Ai empowered wireless networks. *IEEE Communications Magazine*, 57(8):84–90, 2019.

[14] Guo-Wei Lu. Optical signal processing for high-order quadrature- amplitude modulation formats. In Sudhakar Radhakrishnan, editor, *Applications of Digital Signal Processing through Practical Approach*, chapter 1. IntechOpen, Rijeka, 2015.

[15] Ian Marsland, Calvin Plett, and John Rogers. *Radio Frequency System Architecture and Design*. 2013.

[16] Arun Mohan. An overview on regularization, Jan 2019.

[17] Jorge Nocedal and Stephen J Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[18] Nor Noordin, Borhanuddin Ali, S.s Jamuar, Tharek Abd Rahman, and Mahamod Ismail. Adaptive spatial mode of space?time and spacefrequency ofdm system over fading channels. *Jurnal Teknologi*, 45:59–76, 12 2006.

[19] Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw-Hill Education, 2002.

[20] John G. Proakis. *Communication Systems Engineering*. Prentice Hall, 2002.

[21] Theodore S. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2002.

[22] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. *Parallel Distributed Processing, Vol. 1*, pages 318–362, 1985.

[23] M. Shanker, M.Y. Hu, and M.S. Hung. Effect of data standardization on neural network training. *Omega*, 24(4):385–397, 1996.

[24] VK Sharma, MK Singh, and RK Jain. Activation functions in artificial neural networks: a review. *Neural Networks*, 61:87–117, 2014.

[25] Pramila P. Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–6, 2018.

[26] Gilbert Strang. *Linear Algebra and Its Applications*. Wellesley-Cambridge Press, 2019.

[27] Zhiqiang Teng, Shuai Teng, Jiqiao Zhang, Gongfa Chen, and Fangsen Cui. Structural damage detection based on real-time vibration signal and convolutional neural network. *Applied Sciences*, 10:4720, 07 2020.

[28] Wikipedia. 16qam, 2023. `https://uk.wikipedia.org/wiki/16QAM`.

[29] K. K. Wong and R. S. Cheng. *Orthogonal Frequency Division Multiplexing for Wireless Communications: Principles and Practice*. CRC Press, 2017.