A Universal Deep Learning Model for Signal Detection in Wireless Networks

by

Solwin Jabez Solomon,
Dept. of Electrical Engineering and Electronics,
University of Liverpool,
Liverpool, United Kingdom.
sgssolo2@liverpool.ac.uk

Dissertation submitted to the University of Liverpool in partial fulfilment of the requirements for the degree of Master of Science in *Telecommunications and Wireless Systems.*

September, 2025

Acknowledgements

I sincerely thank my supervisor, Dr Huynh Nguyen, for invaluable guidance and support throughout this thesis.

Abstract

Two Important Components in wireless communication systems are signal detection and channel estimation, which have a direct impact on the overall reliability, system performance, and data integrity. In nonlinear, dynamic, or complex propagation scenarios, classical techniques, such as Least Square (LS) and Minimum Mean Square Error (MMSE) detectors, frequently fail to perform effectively due to their heavy reliance on statistical assumptions. Despite the potential of deep learning based solutions to address these challenges, the majority of current models are customised to specific channel conditions or fixed signal-to-noise ratio (SNR) levels, which restricts their adaptability to a variety of real world environments. The objective of this project is to develop a universal deep neural network (DNN) model that is capable of accurately detecting signals and estimating channels in a variety of noise and fading environments. By replacing the components of conventional signal processing infrastructure with a data-driven learning framework, the proposed approach improves the robustness of wireless communication systems. The methodology involves the generation of synthetic datasets using simulated wireless channel models, followed by the design, training, and evaluation of a DNN using TensorFlow and Keras. Quantitative metrics, including Bit Error Rate (BER) and Mean Squared Error (MSE), are implemented to evaluate system performance. To assess enhancements in model generalisation and detection accuracy, a comparative analysis is implemented in comparison to conventional methodologies. Ultimately, the objective of this initiative is to assist in the advancement of intelligent, application oriented communication systems for the deployment of future wireless networks.

Nomenclature

A/D	Analog to Digital
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
CP	Cyclic Prefix
CSI	Channel State Information
D/A	Digital to Analog
DL	Deep Learning
DNN	Deep Neural Network
EM	Expectation Maximization
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
IoT	Internet of Things
LMMSE	Linear Minimum Mean Square Error
LS	Least Squares
MIMO	Multiple Input Multiple Output
ML	Maximum Likelihood
MMSE	Minimum Mean Square Error
NLP	Natural Language Processing
OFDM	Orthogonal Frequency Division Multiplexing
QAM	Quadrature Amplitude Modulation

QPSK	Quadrature Phase Shift Keying
Rx	Receiver
SISO	Single Input Single Output
SNR	Signal to Noise Ratio
Tx	Transmitter

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Wireless connections in the real world are not usually stable. Buildings, cars, people, and equipment problems always affect how a signal moves from a transmitter to a receiver. A receiver that works well in one spot or at one time might have trouble just a moment later. This thesis addresses this situation by creating and testing a Universal deep learning model for orthogonal frequency-division multiplexing (OFDM). This model performs two main tasks: channel estimation and signal detection. It works reliably under various conditions, including additive white Gaussian noise, Rayleigh and Rician fading, and standard profiles like WINNER II, across a practical range of signal-to-noise ratios.

Motivation

I chose this project because my skills and interests fit with a real need in today's networks. I have a bachelor's degree in Electronics and Communication and have worked in the telecom field, specifically with core networks. Working near to the core has taught me how often real links change because of mobility, interference, or hardware restrictions, and how hard it is to keep receivers working effectively in all of these conditions. Telecom companies also demand solutions that are dependable, energy-efficient, and easy to use on a large scale. I'm especially interested in how next-generation technologies (5G and beyond 6G) can provide high data speeds and low latency without making things more complicated to run. This thesis is a step in that direction. I look into a single, universal deep learning receiver that can change channels without having to be retrained every time, instead than making a lot of specialised receivers for different situations. The attraction is useful: one model that works well on the road, from a tranquil, line-of-sight link to a crowded, fast-fading street. This cuts down on engineering work, saves power when it can, and makes the user experience better.

In brief, this project combines my training in communications with my interest in next-generation networks to come up with a design that functions well and is technically solid.

Problem statement

Classical receivers that use models are clear and quick. They use familiar pilot symbols and expected channel statistics to estimate the channel (like least-squares and linear minimum mean-square error) and then equalise each subcarrier (such as zero-forcing and linear minimum mean-square error). These methods work well when their assumptions are true, but their performance decreases if the channel model or interference pattern is different from what was expected.

Traditional deep learning receivers learn from data and can fix some modelling issues, like non-linear hardware effects. Many published designs are trained on just one channel family or a narrow signal-to-noise ratio and then tested under similar conditions. When the deployment environment changes (like moving from Rayleigh to Rician, or from low to high mobility), these models often become less accurate. Gathering data and retraining for each new situation is expensive, slow, and not practical for live systems.

Thesis aim

The objective of this dissertation is to develop, train, and assess a universal deep learning model. A Deep Neural Network that is capable of performing channel estimation and signal detection across a variety of channels and signal-to-noise ratios without the need for retraining in the event of a change in conditions. The receiver is compatible with pilot patterns and demappers that are commonly employed in practice and operates on standard OFDM resource grids.

Why a universal approach now

A universal receiver has three practical benefits. First, **robustness**: training on a purposely mixed dataset allows the model to function even when the channel family or signal-to-noise ratio changes. Second, **operational simplicity**: a single conditional model can replace several scenario-specific pipelines, simplifying maintenance and deployment. Third, **resource efficiency**: if a learnt receiver achieves the same or better bit-error rate with fewer pilots or lighter equalisation, it may reduce processing load and energy use at both the base station and user equipment.

1.1 Wireless communication systems

Wireless communication sends information through the air with radio waves. A transmitter changes bits into a waveform, the waveform moves through the environment, and a receiver attempts to get back the original bits. What makes this difficult is the radio channel. Buildings, vehicles, and even people can reflect and scatter the signal. Movement changes the channel all the time, and the receiver also picks up thermal noise from electronics. Good receivers do two things well: they learn how the channel has changed the signal and they decide which bits were sent.[3, 4, 31].

How the radio channel behaves. Two effects matter most:

- **Large scale effects** explains how the strength of a signal changes as you move farther away and when there are obstacles in the way. Power decreases as distance increases (path loss) and is also lowered in crowded areas (shadowing). Standards groups offer models for these effects in common situations like urban macro, urban micro, and indoor settings. [27].

- **Small scale fading** depicts the quick amplitude changes that result from a combination of several reflected routes. In situations when there is not a strong line-of-sight path, the fading is typically characterised as Rayleigh. For situations where there is a strong direct path, Riccian fading works well. These variations occur rapidly throughout time in fast mobility (Doppler) [28, 29].

As a result of these effects, a single wideband signal might come in ”smeared” over time (delay spread) and can drift in frequency (Doppler and oscillator offsets). Modern systems use defined channel profiles like WINNER II and 3GPP TR 38.901 to simulate and test against these conditions. This makes sure that the findings are realistic and can be compared. [27, 28, 29].

A compact baseband model. It is easy to represent the received complex baseband signal as a simple sum of delayed copies of the transmitted samples plus noise after down-conversion and sampling:

$$r[n] = \sum_{\ell=0}^{L_h-1} h[\ell]\, x[n-\ell] + w[n], \qquad w[n] \sim \mathcal{CN}(0, \sigma^2). \tag{1.1}$$

In this case, $x[n]$ is the sequence that is sent, $h[\ell]$ represents the channel ”taps” that capture multipath effects, L_h is the length of the effective channel, and $w[n]$ is the complex Gaussian noise. The equation in Equation (1.1) is the foundation for the methods discussed in this thesis. It shows why single carrier transmissions experience inter-symbol interference when $L_h>1$. This leads to the use of multicarrier waveforms, like orthogonal frequency-division multiplexing, which simplify the issue into several easier, narrowband subproblems. [3, 4, 31].

Signal quality and how we report it. We use the Signal–to–Noise Rati to quantify the strength of the received signal in relation to the noise. For digital constellations transmitting $\mu = \log_2(M)$ bits per symbol (such as M –ary Quadrature Amplitude Modulation), the established relationship between symbol and bit signal-to-noise ratios is

$$\frac{E_b}{N_0} = \frac{E_s}{N_0} \cdot \frac{1}{\mu}, \tag{1.2}$$

where E_s is average symbol energy and E_b is energy per bit. In experiments we will plot Bit–Error Rate versus Signal–to–Noise Ratio to compare receivers fairly across channels and operating points. Lower Bit–Error Rate at the same Signal–to–Noise Ratio means a better design [3, 4].

1.2 OFDM in wireless communication

Orthogonal frequency division multiplexing(OFDM), is easiest to understand when you see it in action. A lot of current systems use OFDM now, but engineers used single carrier modulation (like Phase Shift Keying and Quadrature Amplitude Modulation on a single wideband carrier) and frequency-division (separate carriers that didn't overlap) before that. Single carrier links are easy to use and don't use much power, but they have a problem called frequency selective fading when they work over wide bandwidths. This happens because different frequencies have different gains and phases, which causes interference between symbols and needs long, adaptable time domain equalisers. Classical spread spectrum and code–division methods make things more reliable in some situations, but their receivers and how they divide up resources don't work as well with today's internet data services. A compromise method called single carrier with frequency domain equalisation makes equalisation easier by using a cyclic prefix and an FFT. However, it still doesn't have the fine subcarrier granularity and variable pilot layouts of OFDM. [3, 4, 31].

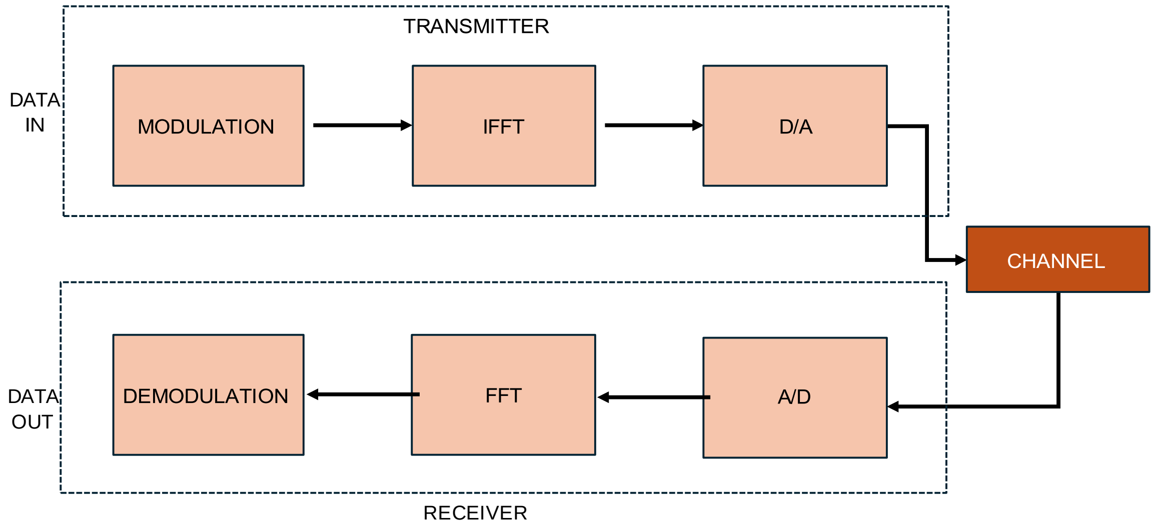


Figure 1.1: Basic OFDM Block Diagram [1]

Why OFDM often wins in practice. OFDM divides a wide band into several narrow, orthogonal subcarriers. Each subcarrier carries a simple symbol, so a "difficult" wideband channel becomes many "easy" flat subchannels. This means: - Equalisation is done with one tap for each subcarrier, making it simple and quick. - Pilots can be arranged on a structured grid for clear channel estimation and tracking. - Schedulers can assign groups of tones to users or services with great detail. - Multiple-antenna techniques like beamforming and spatial multiplexing work well on the subcarrier grid. The benefits of OFDM, along with effective fast Fourier transforms and the cyclic prefix method, led to its popular use as the concepts developed from theory into practical standards. Many 3GPP/WINNER scenarios are set on an OFDM grid with standard pilot patterns, making it the best "testbed" for fair comparisons in this thesis [27, 28, 29].

OFDM baseband model. Let K be the number of subcarriers and X_k the complex symbol (data or pilot) on subcarrier k. One OFDM symbol (without cyclic prefix) is the inverse discrete Fourier transform of $\{X_k\}$:

$$x[n] = \sqrt{\frac{1}{K}} \sum_{k=0}^{K-1} X_k \, e^{j2\pi kn/K}, \qquad n = 0, \dots, K-1. \tag{1.3}$$

A short cyclic prefix copies the tail of $x[n]$ to its front to protect against multipath. After removing the prefix and applying a K-point discrete Fourier transform at the receiver, each subcarrier decouples into a simple "one-tap" relation:

$$Y_k = H_k X_k + W_k, \qquad k = 0, \dots, K-1, \tag{1.4}$$

where H_k is the channel frequency response on tone k and W_k is complex Gaussian noise. Equation (1.4) is the workhorse that enables straightforward pilot-aided channel estimation and per-tone symbol detection used throughout this thesis [3, 4, 7, 8, 9].

How OFDM compares to other modulation choices.

- **Versus single–carrier PSK/QAM:** single–carrier systems need long, adaptive time-domain equalisers when the channel is frequency-selective; OFDM uses a cyclic prefix and reduces equalisation to one complex multiply, which is far simpler and more robust in wideband fading [3, 4].
- **Versus traditional FDM:** classical frequency-division uses guard bands between carriers to avoid interference, which wastes spectrum. OFDM's orthogonality lets subcarriers *overlap* in frequency without interfering, improving spectral efficiency [2, 4].
- **Versus single–carrier with frequency-domain equalisation:** SC-FDE shares the cyclic-prefix/FFT machinery and achieves low peak-to-average power ratio, which suits some uplinks. However, OFDM still offers more flexible pilot patterns and fine subcarrier scheduling that align with today's multi-service traffic [31, 27].

Known limitations. OFDM is not perfect. It has a high peak-to-average power ratio, which stresses power amplifiers; systems use clipping, tone reservation, or pre-distortion to help. It is also sensitive to frequency offsets and phase noise, which disturb orthogonality and create inter-carrier interference; accurate oscillators, synchronisation loops, and modest windowing/filtering reduce the impact. Finally, the cyclic prefix costs a small fraction of throughput, but the simplification it enables usually outweighs this overhead [3, 4]. All of these aspects are captured in standard channel evaluations such as WINNER II and 3GPP TR 38.901 used in this work [27, 28, 29].

Why OFDM suits this thesis: This thesis compares classical channel estimation and detection with conventional deep learning and the proposed universal deep learning model. OFDM's one-tap structure (Eq. (1.4)) makes those comparisons transparent: we can hold the waveform and pilots fixed and focus on how well each method estimates H_k and recovers symbols under realistic channels (AWGN, Rayleigh, Rician, WINNER/3GPP). Because OFDM is the dominant physical-layer waveform in current systems, any improvement is directly relevant to practice [4, 27, 28, 29].

1.3 Traditional methods for channel estimation and signal detection

Channel estimation and signal detection are the two fundamental tasks essential to any dependable wireless receiver. The classical approach tackles these with explicit statistical models and linear algebra tools. Because they are transparent and cheap to run, such methods have powered most commercial OFDM systems. Broadly, they fall into three types:

- **pilot-based**,
- **blind**,
- **semi-blind.**

This section reviews the standard pilot aided methods used in orthogonal frequency–division multiplexing receivers for channel estimation and signal detection. The focus is on least squares (LS) and linear minimum mean–square–error (LMMSE) estimators on the pilot grid, and on zero–forcing (ZF) and linear minimum mean–square–error (MMSE) per subcarrier equalisers. These techniques form the classical baselines against which learning based receivers are compared later [3, 4] [7]-[12].

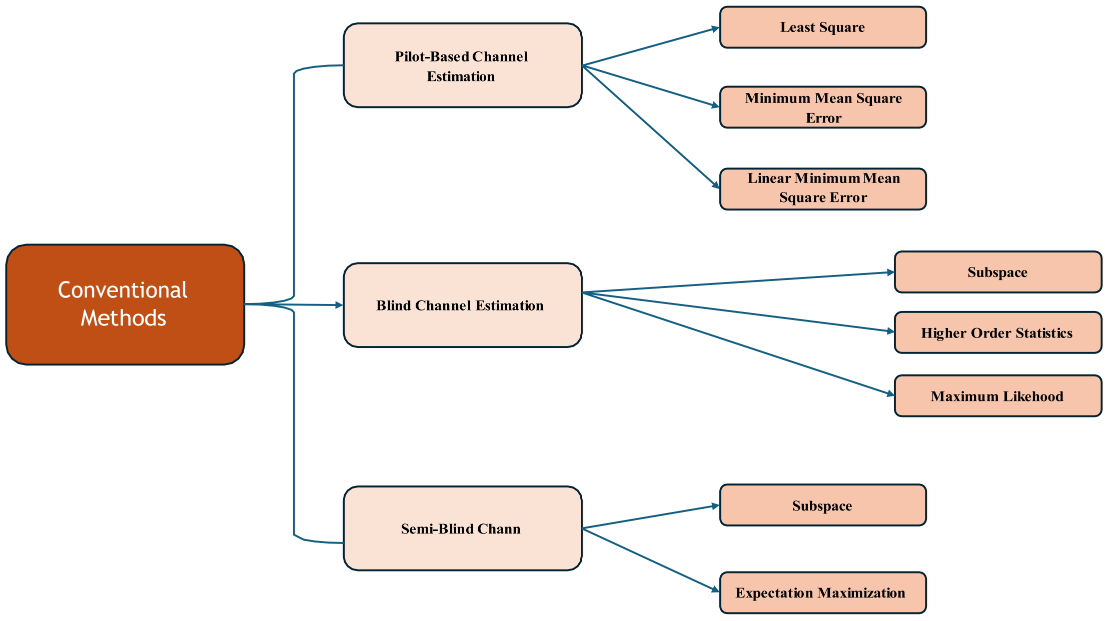


Figure 1.2: Classification of conventional channel estimation methods in wireless Communication Systems [1, 2]

Signal model on the pilot grid. Let K be the number of subcarriers and $\mathrm{P} \subset \{0, \ldots, K-1\}$ the pilot indices. On a pilot tone $k \in \mathrm{P}$ the frequency–domain observation satisfies

$$Y_k = H_k X_k + W_k, \tag{1.5}$$

where X_k is a known pilot symbol, H_k is the channel frequency response, and $W_k \sim \mathcal{CN}(0, \sigma^2)$ is noise [4]. Stacking all pilot tones gives

$$\mathbf{y}_{\mathrm{P}} = \mathbf{D}_{\mathrm{P}} \mathbf{h}_{\mathrm{P}} + \mathbf{w}_{\mathrm{P}}, \qquad \mathbf{D}_{\mathrm{P}} = \mathrm{diag}(\{X_k\}_{k \in \mathrm{P}}), \tag{1.6}$$

with $\mathbf{h}_{\mathrm{P}}$ the vector of $\{H_k\}_{k \in \mathrm{P}}$.

Least–squares channel estimation

Dividing the received pilots by the known pilots yields the per–tone LS estimate

$$\hat{H}_k^{\mathrm{LS}} = \frac{Y_k}{X_k}, \qquad k \in \mathrm{P}, \tag{1.7}$$

which is unbiased and has error variance

$$\mathrm{E}\left[|\hat{H}_k^{\mathrm{LS}} - H_k|^2 \right] = \frac{\sigma^2}{|X_k|^2}. \tag{1.8}$$

In practice the pilots have constant modulus (e.g., quadrature phase–shift keying), so (1.8) reduces to σ^2/E_p with $E_p = |X_k|^2$ [7, 11, 12]. Channel values on data tones $\mathbb{D}$ are then obtained by interpolation across frequency (and time when multiple symbols are used), for example by linear or spline interpolation [7, 11].

LMMSE channel estimation (interpolation with statistics)

When second order channel statistics are known or can be approximated from a power delay profile, Wiener filtering improves over pure interpolation. Writing $\hat{\mathbf{h}}_{\mathrm{P}}^{\mathrm{LS}}$ for the LS pilot estimates, one canonical LMMSE interpolator is

$$\hat{\mathbf{h}}_{\mathrm{D}}^{\mathrm{LMMSE}} = \mathbf{R}_{DP}\left(\mathbf{R}_{PP} + \frac{\sigma^2}{E_p}\mathbf{I}\right)^{-1}\hat{\mathbf{h}}_{\mathrm{P}}^{\mathrm{LS}}, \tag{1.9}$$

where $\mathbf{R}_{PP} = \mathrm{E}[\mathbf{h}_{\mathrm{P}}\mathbf{h}_{\mathrm{P}}^{\mathrm{H}}]$ and $\mathbf{R}_{DP} = \mathrm{E}[\mathbf{h}_{\mathrm{D}}\mathbf{h}_{\mathrm{P}}^{\mathrm{H}}]$ are the pilot–pilot and data–pilot correlation matrices induced by the assumed power–delay profile [7, 8, 9]. For time–varying channels observed over several OFDM symbols, a two–dimensional LMMSE can be built with separable time–frequency correlation $\mathbf{R} \approx \mathbf{R}_t \otimes \mathbf{R}_f$, using the sampling operator $\mathbf{S}$ that selects the pilot positions:

$$\hat{\mathbf{h}} = \mathbf{R}\mathbf{S}^{\mathrm{H}}\left(\mathbf{S}\mathbf{R}\mathbf{S}^{\mathrm{H}} + \frac{\sigma^2}{E_p}\mathbf{I}\right)^{-1}\mathbf{y}_{\mathrm{P}}. \tag{1.10}$$

Pilot layout (comb, block, preamble), spacing, and power allocation control the estimation error versus overhead trade–off and have been studied extensively for orthogonal frequency–division multiplexing [13, 14, 15, 16, 17].

Per–subcarrier signal detection (ZF and MMSE)

With channel estimates $\hat{H}_k$ on all tones, per tone equalisation recovers soft symbol estimates. The zeroforcing rule inverts the channel estimate directly:

$$\tilde{X}_k^{\mathrm{ZF}} = \frac{Y_k}{\hat{H}_k}. \tag{1.11}$$

This removes channel distortion but may amplify noise when $|\hat{H}_k|$ is small. The linear MMSE equaliser balances inversion against noise enhancement:

$$\tilde{X}_k^{\mathrm{MMSE}} = \frac{\hat{H}_k^*}{|\hat{H}_k|^2 + \sigma^2/E_s}\,Y_k, \tag{1.12}$$

where E_s is the average symbol energy [3, 4]. Under perfect channel knowledge, (1.12) minimises $\mathrm{E}[|X_k - \tilde{X}_k|^2]$; with estimation error $\tilde{H}_k = H_k - \hat{H}_k$, the post–equaliser signal–to–interference–plus–noise ratio is well approximated by

$$\gamma_k^{\mathrm{ZF}} \approx \frac{|H_k|^2 E_s}{\sigma^2 + E_s\,\mathrm{E}[|\tilde{H}_k|^2]}, \qquad \gamma_k^{\mathrm{MMSE}} \geq \frac{|H_k|^2 E_s}{|H_k|^2 E_s + \sigma^2}, \tag{1.13}$$

highlighting the robustness of MMSE to both noise and moderate estimation error [3, 4, 24]. After equalisation, soft or hard demapping converts $\tilde{X}_k$ into bit estimates according to the chosen constellation.

Discussion and limitations

LS is attractive for its simplicity and unbiasedness, but it is noise limited and relies on interpolation quality. LMMSE exploits second–order statistics to reduce error variance but requires a channel covariance model and matrix inversions whose size grows with pilot count [7, 8, 9]. For detection, zero–forcing can suffer at spectral nulls while MMSE trades interference removal for noise robustness [3, 4]. Performance degrades when the assumed statistics (e.g., power delay profile, Doppler) depart from reality or when carrier frequency offset and phase noise introduce strong inter–carrier interference, motivating more adaptive approaches evaluated later in this thesis [27, 28, 29].

1.4 Deep learning for channel estimation and signal detection

Deep learning offers adaptable functionality capable of acquiring residual structures and hardware deficiencies from data, while maintaining the orthogonal frequency-division multiplexing framework that facilitates fast per-subcarrier processing. The literature shows two broad benefits that are directly relevant to this thesis: (i) reduced model mismatch in channel estimation and (ii) improved symbol and bit detection under practical impairments [34]-[37][40, 44, 50, 52].

Problem setup. Let **Z** denote features available to the receiver (e.g., pilot subcarriers, noisy data subcarriers, or short time–frequency patches). A *channel estimation network* predicts the frequency response on all tones,

$$\hat{\mathbf{H}} = f_{\boldsymbol{\theta}}(\mathbf{Z}), \tag{1.14}$$

and is trained to minimise a mean–square error objective,

$$\mathcal{L}_{\mathrm{CE}}(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \left\| \mathbf{H}^{i} - f_{\boldsymbol{\theta}}(\mathbf{Z}^{i}) \right\|_2^2 + \lambda \|\boldsymbol{\theta}\|_2^2, \tag{1.15}$$

where λ controls ℓ_2 regularisation [37, 38, 42, 43, 44]. A *signal detection network* maps features to bit probabilities,

$$\hat{\mathbf{p}} = \sigma\left(g_{\boldsymbol{\phi}}(\mathbf{Z})\right), \qquad \hat{p}_b \in (0, 1), \tag{1.16}$$

and uses a (multi–label) binary cross–entropy loss

$$\mathcal{L}_{\mathrm{DET}}(\boldsymbol{\phi}) = -\frac{1}{N}\sum_{i=1}\sum_{b=1}\Big[b_{i,b}\log \hat{p}_{i,b} + (1-b_{i,b})\log(1-\hat{p}_{i,b})\Big], \tag{1.17}$$

optionally combined with (1.15) in a joint objective $\mathcal{L} = \alpha\mathcal{L}_{\mathrm{CE}} + \beta\,\mathcal{L}_{\mathrm{DET}}$ [37, 40, 44, 51]. Here B is the number of information bits per orthogonal frequency–division multiplexing block.

Fully data–driven discriminative models

Fully connected and convolutional networks immediately learn the mappings in (1.14) or (1.16) from labelled samples. Channel estimation use pilot tones as inputs to generate estimates, whereas signal detection employs equaliser outputs, soft information, or raw time frequency patches as inputs. These models include non-linearities including amplifier distortion, quantisation, and oscillator phase noise, while maintaining compatibility with typical demappers [37, 38, 40, 42, 43, 44, 50]. Architectures designed for orthogonal frequency-division multiplexing (e.g., ComNet) integrate straightforward expert layers with learnt residual blocks to enhance resilience while maintaining moderate complexity [44]. Complex valued variations process in-phase and quadrature simultaneously and have demonstrated usefulness for detection in index-modulated orthogonal frequency-division multiplexing [51].

A prevalent method involves overseeing channel estimation using (1.15) and supervising bit detection using (1.17), while assessing bit-error rate against signal-to-noise ratios. Mixed SNR training enhances generalisation across all channels. (utilised thereafter in this thesis for the global network) [37, 40, 50].

Model–driven or physics–informed networks

Networks with learnable step sizes or shrinkage functions are integrated into algorithms through algorithm unrolling. These networks may contain several iterations of classical estimators, such as Wiener filtering or an approximate message passing step.

$$\mathbf{z}^{(t+1)} = \eta_{\boldsymbol{\theta}_t}\left(\mathbf{z}^{(t)} + \mathbf{B}_t\,\mathbf{y}\right), \qquad t = 0, \ldots, T-1, \tag{1.18}$$

where $\eta_{\boldsymbol{\theta}_t}$ are light nonlinearities and $\mathbf{B}_t$ are learnable linear operators. This reduces the need for sample size, preserves inductive bias, and produces interpretable modules with a complexity that is comparable to model-based baselines [33, 44]. In orthogonal frequency–division multiplexing receivers, such hybrids frequently implement a learnt refinement subsequent to a pilot-aided initial estimate, thereby enhancing the mean–square error and bit–error rate in the event of a mismatch to the assumed statistics [44, 42].

Sequence and attention models

Temporal correlation among orthogonal frequency-division multiplexing symbols and frequency correlation can be utilised with recurrent or attention layers. Recurrent units operate as

$$\mathbf{s}_t = \mathrm{RNN}_{\boldsymbol{\theta}}(\mathbf{s}_{t-1}, \mathbf{z}_t), \qquad \hat{\mathbf{H}}_t = \mathbf{W}\mathbf{s}_t, \tag{1.19}$$

capturing slow channel evolution for improved tracking and detection [47, 45, 46]. Transformer encoders use multi–head attention to fuse long–range dependencies:

$$\mathrm{Attn}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right)\mathbf{V}, \tag{1.20}$$

and have been adopted to model time–frequency contexts that are hard to capture with local filters alone, improving robustness to Doppler and frequency–selective fading [54, 41].

Self–supervised and semi–supervised training

In many wireless scenarios, the number of labelled pilots or channel samples are limited. Self–supervised learning provides a way to stabilise training by enforcing consistency between different views of the same signal. For example, two augmentations aug_1 and aug_2 of the same received symbol block (such as jittered pilot positions or outputs from a stochastic equaliser) are encouraged to produce similar predictions:

$$\mathcal{L}_{\mathrm{cons}}(\boldsymbol{\theta}) = \left\| f_{\boldsymbol{\theta}}(\mathrm{aug}_1(\mathbf{Z})) - f_{\boldsymbol{\theta}}(\mathrm{aug}_2(\mathbf{Z})) \right\|_2^2. \tag{1.21}$$

Semi-supervised algorithms may also make pseudo-labels from predictions that are rather convinced of themselves, which can help the training set. These methods work together to make the model less reliant on dense pilot grids and better equipped to handle a range of channel conditions. [35, 36, 44].

What prior results suggest

Evidence from previous studies shows that (i) deep learning can reduce mean–square error in channel estimation, when the assumed statistics are inaccurate, and (ii) learned detectors can close the gap to ideal demappers under hardware non–idealities and interference [37]-[44],[50, 51, 52]. These observations motivate the universal deep neural network designed in later chapters.

1.5 Universal deep learning for signal detection and channel estimation

Most deep learning receivers are trained on a single, restricted channel, such as one fading family and a small signal-to-noise range, and then tested under the same settings. In practice, wireless links can be very different. For example, channels can be Rayleigh, Rician, line-of-sight with sparse scatterers, or standardised WINNER/3GPP profiles with different delay and Doppler spreads. The operating point can also change depending on mobility, hardware problems, and interference [27, 28, 29]. When the circumstances for deployment change, models that were trained on just one distribution generally become less accurate. This means that channel estimations are less accurate and bit error rates are higher [34, 35, 37, 52]. It is not a good idea to gather data and retrain for every new situation.

A universal deep learning receiver offers a practical alternative. The objective is to develop a single model that is consistent across a variety of channel families and signal-to-noise ratios without the need for retraining. A designed mixture is used to select mini-batches during training. This combination includes standardised scenarios from WINNER/3GPP, Rayleigh and Rician fading, and additive white Gaussian noise. This varied curriculum promotes the network's acquisition of features and decision rules that are applicable to multiple conditions, as opposed to overfitting to a single case [34, 37, 52]. In the field, the model does not require switching or fine-tuning, as the same weights are able to adapt to the current operating point through lightweight conditioning (e.g., passing the pilot mask, an approximate SNR estimate, or the modulation order into small side networks). A simple universal deep neural network (DNN) architecture for channel estimation and signal detection is illustrated below.

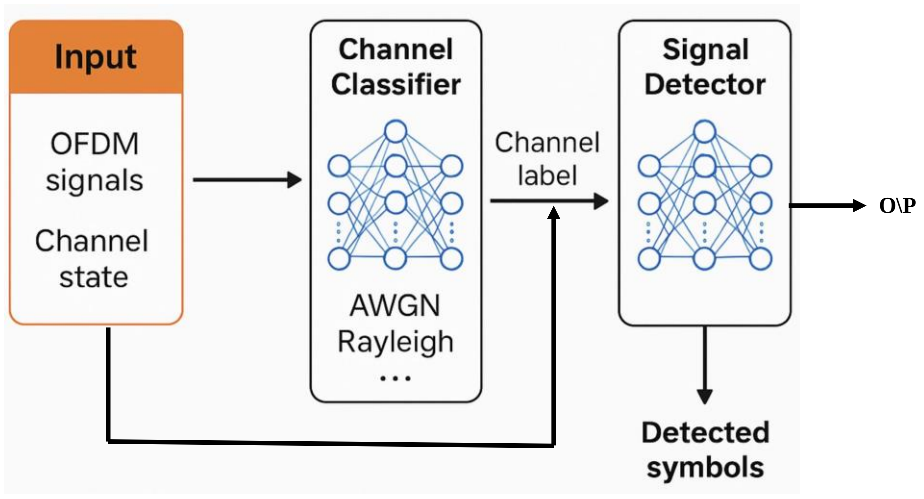


Figure 1.3: simple Universal DNN Model

A universal receiver makes sense from an operational standpoint due to its ease of deployment and maintenance. The cost and engineering overhead are reduced as a result of the replacement of multiple scenario specific networks by a single conditional model. It can decrease its dependency on compact pilots, which in turn reduces overhead and power consumption, by learning how to leverage the structure in the pilot layout and the time–frequency domain. The real-time demonstrations of compact deep learning orthogonal frequency–division multiplexing receivers indicate that these models can satisfy practical latency and throughput constraints when carefully constructed [50]. In summary, universal deep learning aims to preserve the transparency and efficiency of conventional practice to the greatest extent possible while utilising data-driven learning to provide reliable channel estimation and signal detection in dynamic wireless environments [34, 37, 40, 44, 52].

Chapter 2

Literature Review

2.1 Conventional signal detection techniques

Signal detection and channel estimation are core building blocks of reliable wireless receivers. Traditional designs rely on explicit statistical models and linear algebra, and they have powered many commercial orthogonal frequency–division multiplexing systems thanks to their clarity and low computational cost [3, 4]. In the literature, conventional approaches are commonly grouped into three families: *pilot–based*, *blind*, and *semi–blind* methods [7].

Pilot–based methods. Recognised pilot symbols are included into a time-frequency grid. A least-squares estimation of the pilot tones depends on dividing the received pilots by the broadcast pilots; this method is relatively simple and unbiased, however it exhibits significant sensitivity to noise [7, 11, 12]. Extensive research has been conducted on the conversion from pilot tones to data tones (including comb, block, or preamble configurations), since the arrangement and density of pilots significantly influence accuracy and overhead [13]-[23]. To enhance robustness, linear minimum mean-square-error filtering utilises second-order channel statistics, such as the power-delay profile, and generally surpasses least-squares methods at moderate and low signal-to-noise ratios, albeit with the drawbacks of matrix inversions and dependence on an accurate correlation model [7, 8, 9]. Following to channel estimation, per-subcarrier zero-forcing or linear MMSE equalisation yields soft symbols for demapping [3, 4].

Blind methods. Blind estimate eliminates explicit pilots and utilises the structure or statistics of the transmitted signal. Prominent methodologies including subspace techniques reliant on second-order statistics, higher-order statistics, and maximum-likelihood formulations [7]. These techniques reduce pilot overhead but frequently need extended observation intervals, are vulnerable to initialisation and model discrepancies, and may prove unreliable in fast fading or low signal-to-noise conditions [7].

Semi–blind methods. Semi-blind approaches integrate a limited number of pilots with statistical inference to optimise accuracy and spectral efficiency. Conventional models combine sparse pilots with subspace estimates and optimise parameters by expectation-maximization, alternating between soft symbol decisions and channel updates [7]. Although they minimise pilot load compared to entirely pilot-based designs, their effectiveness remains dependant upon the presumed correlation models and could decline under significant Doppler effects, carrier frequency offsets, or hardware limitations [3, 7].

Limitations and motivation for Deep learning. All three approaches depend on established modelling assumptions. When deployed channels diverge from these assumptions due to mobility, obstruction, hardware imperfections, or a transition among standardised situations (e.g., WINNER/3GPP) the estimators may become distorted or exhibit reduced efficiency [27, 28, 29]. Enhancing resilience generally elevates pilot overhead or computational expenses (for instance, extensive LMMSE filters), hence reducing spectral efficiency or complicating real time processing [7, 23, 24, 25]. These constraints need data-driven receivers that adapt to rectify model discrepancies while maintaining compatibility with the OFDM framework. Previous research has documented improvements in channel estimation and detection amongst practical drawbacks, hence motivating the subsequent evaluation of deep learning receivers and the universal model suggested in this thesis. [34, 35, 37, 40, 44, 50, 52].

2.2 Conventional deep learning for channel estimation and signal detection

2.2.1 Introduction to conventional deep learning

Deep learning has been widely explored to improve channel estimation and signal detection in orthogonal frequency–division multiplexing systems. The main idea is to learn mappings from pilot and noisy data observations to either channel responses or bit decisions, thereby addressing modelling errors (e.g., imperfect power delay profiles, residual phase noise, synchronisation) that limit classical linear methods. Early physical layer surveys and tutorials established the feasibility of deep learning modules in the receiver chain and highlighted opportunities for gains under practical impairments [34, 35, 38]. Subsequent studies developed networks to orthogonal frequency–division multiplexing, demonstrating that learned estimators and detectors can reduce mean–square error and improve bit error rate compared with least–squares and linear minimum mean–square–error baselines when channel statistics are uncertain or time varying [37],[40]-[44], [50, 51, 52].

2.2.2 Types of deep learning architectures

Fully connected neural networks. Fully connected neural networks consist of stacked interconnected layers with non-linear activations and are the simplest deep models to insert

H

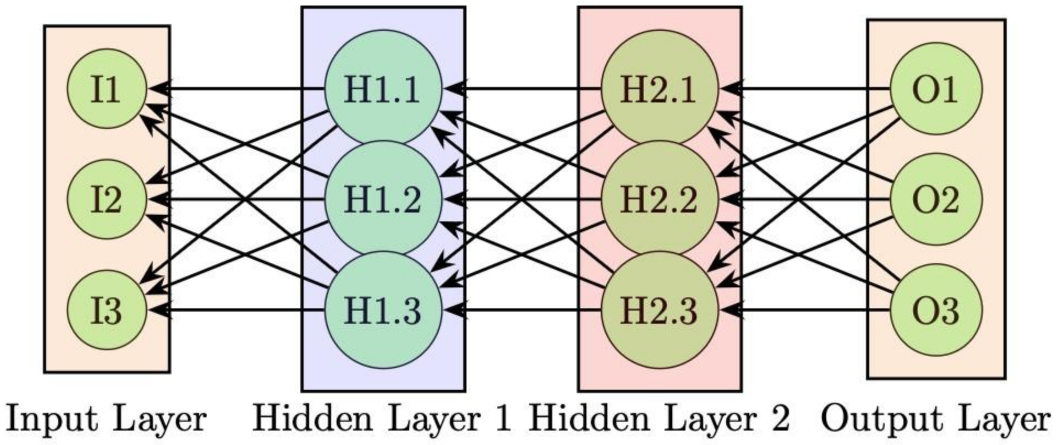


Figure 2.1: Fully connected Neural Network[44]

into an orthogonal frequency–division multiplexing receiver. They map a fixed length feature vector to a desired output such as refined channel coefficients on selected subcarriers or hard/soft bit decisions without assuming locality or convolutional structure. In practice, complex valued inputs (in-phase and quadrature) are arranged as two real channels and concatenated with added signals. [37, 40, 42, 51].

Convolutional Neural Networks. Convolutional Neural Networks operate on local time–frequency patches of the orthogonal frequency–division multiplexing resource grid. Weight sharing across frequency and time captures neighbourhood structure and reduces parameter count. In receiver design, encoder–decoder variants take pilots (and nearby tones) as input and output full-grid channel estimates, or they refine equaliser outputs before demapping. Architectures that combine expert layers with convolutional residual blocks (e.g., an initial pilot-aided estimate followed by a learned refinement) have been shown to improve robustness with modest complexity [44, 42, 43, 37].

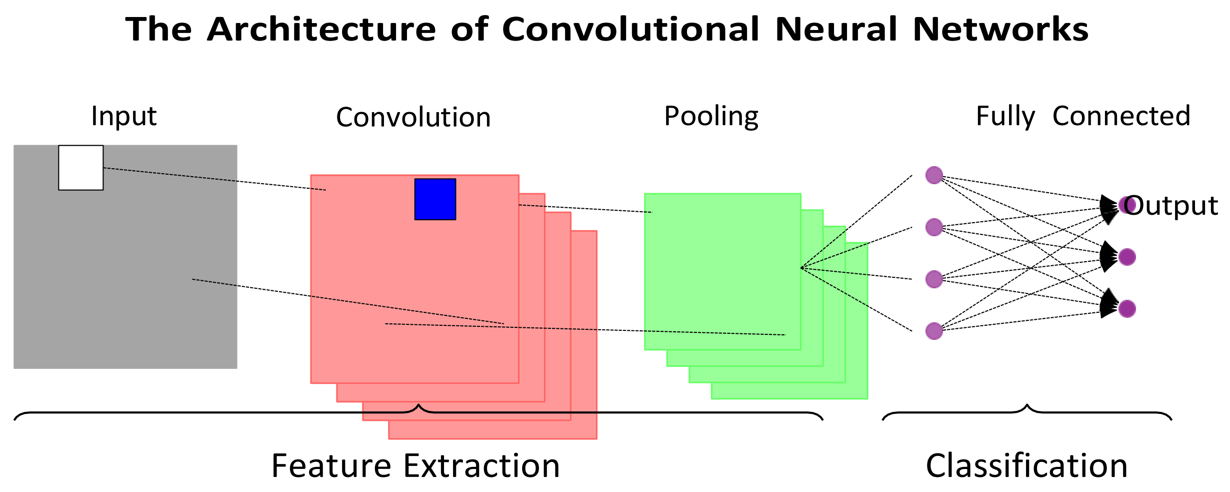


Figure 2.2: Convolution Neural Network[3]

Deep Neural Networks. Deep Neural Networks with fully connected layers are often used when features are low dimensional (e.g., per-subcarrier stacks of pilot and equaliser statistics) or when complex-valued processing can be represented as two real channels. They are simple to train and integrate, and have been applied to both channel estimation and symbol detection, showing improved performance over linear rules in settings with non-linear distortions or coarse statistical knowledge [37, 40, 42, 51]. Complex-valued extensions further exploit in-phase and quadrature structure for detection in index-modulated systems [51].

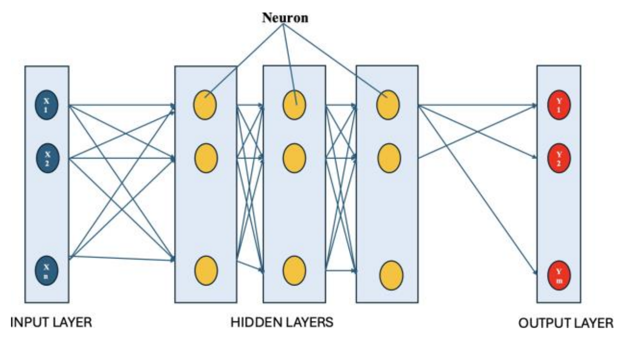


Figure 2.3: Deep Neural Networks[37]

2.2.3 Efficiency compared to traditional methods

Compared with least–squares and linear minimum mean–square–error pipelines, conventional deep learning architectures report three recurring advantages:

1. **Accuracy under mismatch.** When the assumed power–delay profile, Doppler spectrum, or hardware model is inaccurate, learned estimators and detectors reduce mean–square error and bit error rate relative to linear baselines by capturing residual non-linearities and coupling across tones [37, 40, 41, 42, 43, 44, 52].

2. **Computational efficiency at inference.** Once trained, compact convolutional or shallow fully connected networks execute as a small number of dense or convolutional layers. Prototyping results show real-time feasibility on embedded hardware and field-programmable gate arrays, indicating that inference complexity can be comparable to classical interpolation and per-tone equalisation [50].

3. **Adaptability in practical pipelines.** Because these models operate on the orthogonal frequency–division multiplexing grid, they integrate with standard pilot patterns and demappers without redefining the waveform. Hybrid designs that start from a pilot-aided estimate and apply a learned refinement offer a low-risk upgrade path [44, 42, 43].

At the same time, conventional deep learning systems have limitations that motivate broader generalisation: many are trained on a single channel family and a narrow signal–to–noise range, which can reduce reliability when deployment conditions shift; data labelling relies on simulators with specific assumptions; and re-training across scenarios increases operational burden [34, 35, 52]. These issues set the stage for a universal approach that trains across channel families and operating points and conditions the network on lightweight context, as proposed in this thesis.

2.2.4 Summary

Representative contributions over the last decade include the following.

- **Deep learning for the physical layer.** Tutorials and surveys established that deep learning modules can be inserted into receiver chains and may outperform analytical rules under practical impairments, while outlining open challenges in robustness and data efficiency [34, 35, 38, 32].

- **Channel estimation and detection in orthogonal frequency–division multiplexing.** A joint estimator–detector demonstrated that learning from pilots and noisy tones can improve mean–square error and bit error rate over least–squares and linear minimum mean–square–error under Rayleigh fading and oscillator imperfections [37]. Follow-up works designed specialised estimators for doubly selective channels [38], high-speed scenarios [43], and joint channel estimation with signal detection [40], each reporting gains over linear interpolation or Wiener filtering under their respective assumptions.

- **Hybrid (expert + learned) receivers.** Combining classical signal processing with learned residual blocks (e.g., ComNet) yields interpretable receivers that preserve per-tone structure while correcting residual mismatch; these hybrids consistently improved detection accuracy at modest computational cost compared with purely linear baselines [44, 42].

- **Sequence and attention models.** Works using recurrent and attention-based encoders showed improved tracking of time-varying channels and better robustness to Doppler compared with frame-wise processing, supporting the use of temporal context in receivers [41, 45, 46, 47].

- **Practicality and real-time deployment.** A real-time deep learning orthogonal frequency–division multiplexing receiver demonstrated that compact models meet latency and throughput constraints on hardware, strengthening the case for deployment in commercial-type settings [50].

Conventional deep learning has delivered consistent gains over least–squares and linear minimum mean–square–error baselines by learning from pilots and local time–frequency structure, using Convolutional Neural Networks, Deep Neural Networks, Recurrent Neural Networks, and model-driven unrolled designs [37, 40, 41, 42, 43, 44]. However, most published models are trained for a single channel type or a narrow operating range and can lose accuracy when the environment changes [34, 35, 52]. This limitation motivates the universal deep learning approach developed in this thesis, which trains across diverse channel families and signal–to–noise ratios and conditions on lightweight context to maintain performance without retraining.

Chapter 3

Channel Models and Metrics

3.1 Propagation channel models for evaluation

This section defines the *radio channels* used to test all receivers in this thesis. Each model represents a different real-world condition: richly scattered streets with no clear line-of-sight, links with a dominant path, standardized multi-path profiles, and a clean noise-only baseline. We keep the number of equations small and focus on what each model means for orthogonal frequency–division multiplexing processing and Bit–Error Rate reporting [3, 4].

3.1.1 Rayleigh fading channel

Intuition. Rayleigh fading describes environments with many reflections and *no* dominant line-of-sight path (e.g., dense urban streets). Signal strength can dip sharply and change from subcarrier to subcarrier and from symbol to symbol.

Minimal model. A narrowband complex channel coefficient can be written as a zero-mean circular complex Gaussian variable,

$$h \sim \mathcal{CN}(0, 2\sigma^2), \qquad |h| \text{ is Rayleigh}, \qquad |h|^2 \text{ is exponential}, \tag{3.1}$$

so each orthogonal frequency–division multiplexing tone experiences a random scale and phase. For coherent detection of Quadrature Phase Shift Keying with perfect tracking, a useful reference for the average uncoded bit–error rate in Rayleigh is

$$P_b^{\text{QPSK,Rayleigh}} = \frac{1}{2}\left(1 - \sqrt{\frac{\gamma}{1+\gamma}}\right), \quad \gamma = E_b/N_0, \tag{3.2}$$

which we use only as a baseline to interpret simulation curves [3, 4]. In practice, pilot density and estimator quality determine how close a receiver gets to this reference.

3.1.2 Rician fading channel

Intuition. Rician fading adds a *dominant* component (often a line-of-sight or strong specular reflection) on top of diffuse scattering. As that dominant path strengthens, fading becomes milder and decisions are more stable.

Minimal model. A convenient decomposition is

$$h = m + g, \qquad g \sim \mathrm{CN}(0, 2\sigma^2), \tag{3.3}$$

with the *Rician K-factor*

$$K = \frac{|m|^2}{2\sigma^2}, \tag{3.4}$$

measuring the power ratio of the dominant to scattered components. Larger K implies less severe fades and typically lower Bit–Error Rate at the same Signal–to–Noise Ratio; $K \to 0$ recovers the Rayleigh case [3, 4]. Robust receivers should degrade gracefully as K decreases.

3.1.3 WINNER II stochastic channel models

Intuition. WINNER II provides standardized, measurement-based *tapped-delay-line* profiles for typical scenarios (e.g., urban macro/micro, indoor). These models specify realistic power–delay profiles, angle spreads, and Doppler characteristics so that evaluations are comparable across studies [33, 39].

Minimal frequency-domain view. For an orthogonal frequency–division multiplexing tone at frequency f_k, the frequency response is the superposition of P paths with delays τ_p and complex gains $a_p(t)$:

$$H(f_k, t) = \sum_{p=1} a_p(t)\, e^{-j2\pi f_k \tau_p}, \tag{3.5}$$

where the set $\{\tau_p, a_p(t)\}$ is drawn according to the chosen WINNER II scenario (cluster powers, delays, and Doppler spectra). This compact form highlights what the receiver must estimate on each subcarrier while respecting the scenario's time variation and frequency selectivity [33, 39]. In this thesis we use WINNER II profiles only (no 3GPP TR 38.901 extensions).

3.1.4 Additive White Gaussian Noise (AWGN) baseline

Intuition. AWGN is the cleanest baseline: there is *no fading*, only thermal noise from electronics. It is used to sanity-check mappings, demappers, and our Bit–Error Rate versus Signal–to–Noise Ratio reporting.

Minimal model. Each tone follows a one-tap relation with noise,

$$Y_k = X_k + W_k, \qquad W_k \sim \mathrm{CN}(0, N_0), \tag{3.6}$$

so differences among receivers should be small at moderate-to-high Signal–to–Noise Ratio if implementations are correct [3, 4]. We include AWGN curves mainly to validate and calibrate the simulation pipeline.

Notes for interpretation. Across these models, we keep the orthogonal frequency–division multiplexing grid, pilot pattern, and constellations (Quadrature Phase Shift Keying and 16–Quadrature Amplitude Modulation) fixed so that comparisons among least–squares, conventional deep learning, and the proposed universal Deep Neural Network are fair and focused on receiver design rather than waveform changes.

3.2 Constellation Mapping for QPSK and QAM

Modulation in wireless communication is the process of mapping digital bits into complex-valued symbols for transmission. Quadrature Phase Shift Keying (QPSK) and Quadrature Amplitude Modulation (QAM) are widely used in OFDM systems because they balance efficiency and robustness.

Quadrature Phase Shift Keying (QPSK): Imagine a compass with four directions—north, south, east, and west. In QPSK, each direction represents two bits, and the information is carried only by the *phase* of the signal. Because all constellation points lie on a circle with equal energy, QPSK is resilient against noise and is often used in low-to-moderate signal-to-noise ratio (SNR) conditions [3, 4]. Mathematically, QPSK maps two input bits (b_0, b_1) into a complex symbol as

$$X_{\text{QPSK}} = \sqrt{\frac{E_s}{2}}\,(2b_0-1)+j(2b_1-1)\,, \tag{3.7}$$

where E_s is the average symbol energy.

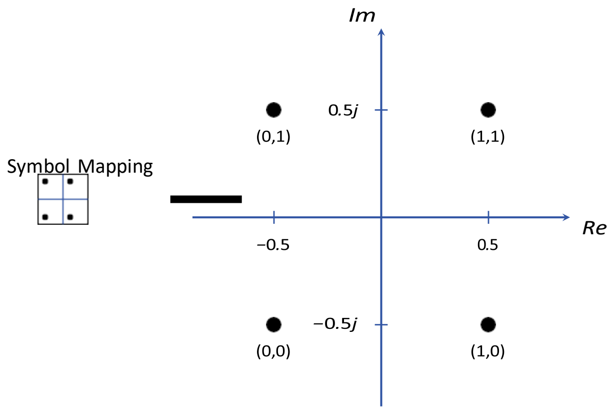


Figure 3.1: QPSK Constellation Mapping[3, 4]

Quadrature Amplitude Modulation (QAM): Unlike QPSK, which uses only phase, QAM encodes information in both amplitude and phase. For example, in 16-QAM, there are 16 points arranged on a square grid, with each point carrying 4 bits. This allows higher spectral efficiency but reduces the distance between points, making QAM more sensitive to noise and fading [3, 4, 27]. The general mapping for square M-QAM is

$$X_{\text{QAM}} = \alpha(a + jb), \quad a, b \in \{\pm 1, \pm 3, \ldots, \pm(\sqrt{M} - 1)\}, \tag{3.8}$$

where α is a scaling factor chosen to normalise the average energy of the constellation to E_s. For 16-QAM, the set $\{\pm 1, \pm 3\}$ is used, yielding higher data rates but at the expense of reduced robustness [5].

In summary, QPSK provides robustness with fewer bits per symbol, while higher-order QAM (e.g., 16-QAM) increases throughput but requires higher SNR for reliable detection. Adaptive modulation schemes in wireless standards often switch between these depending on the channel quality [27, 28, 29].

Square M-QAM (with emphasis on M=16)

For square M-QAM, the in-phase and quadrature levels are drawn from $\mathcal{A} = \{\pm 1, \pm 3, \ldots, \pm(\sqrt{M} - 1)\}$ With standard normalisation,

$$X_{\text{QAM}} = \alpha\,(a + jb), \quad a, b \in \mathcal{A}, \qquad \alpha = \sqrt{\frac{E_s}{\frac{2}{3}(M-1)}}, \tag{3.9}$$

so that $\mathbb{E}[|X_{\text{QAM}}|^2] = E_s$ [3, 4]. For M =16, $\mathcal{A} = \{\pm 1, \pm 3\}$ and $\alpha = \sqrt{E_s/10}$. Higher M increases spectral efficiency ($\log_2 M$ bits/symbol) but reduces the minimum distance between points, raising error sensitivity in noise and fading.

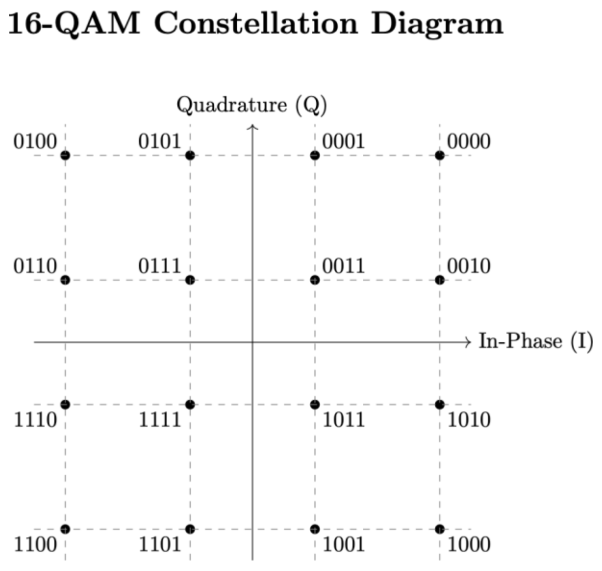


Figure 3.2: 16 QAM Constellation Mapping[3, 4]

3.1.3 Signal-to-Noise Ratio (SNR)

Signal-to-Noise Ratio is a fundamental performance metric in wireless communication that quantifies the relative strength of the transmitted signal compared to background noise. In orthogonal frequency–division multiplexing (OFDM) systems, SNR directly determines the reliability of symbol detection because it affects how clearly constellation points such as Quadrature Phase Shift Keying (QPSK) or Quadrature Amplitude Modulation (QAM) can be distinguished at the receiver. A higher SNR implies cleaner reception and lower probability of symbol errors, while a lower SNR results in overlapping constellation points and higher detection errors [3, 4, 27]. In practice, adaptive modulation and coding schemes use SNR thresholds to decide whether to transmit with robust QPSK or switch to higher-order QAM for greater data rates [28, 29]. Therefore, accurate estimation of SNR is essential for both system evaluation and real-time adaptation in modern wireless standards.

3.1.4 Bit-Error Rate (BER)

Bit-Error Rate measures the proportion of incorrectly decoded bits relative to the total transmitted bits and is one of the most widely used performance indicators in communication systems. In OFDM, BER reflects the combined influence of channel fading, noise, interference, and the accuracy of channel estimation. For example, in slow-fading channels with sufficient pilot density, BER is primarily limited by additive noise, while in fast-varying channels, it also captures errors due to outdated or inaccurate channel estimates [7, 8, 9]. BER curves plotted against SNR provide a direct benchmark of modulation efficiency and detection performance. Quadrature Phase Shift Keying typically achieves lower BER at low SNR due to its wider decision boundaries, whereas 16-QAM achieves better spectral efficiency but is more error-prone under fading [37, 41, 43]. BER analysis is thus a critical tool in this thesis for comparing traditional and deep learning based detection techniques across multiple channel models.

3.1.5 Payload bits and pilot bits

In an OFDM frame, the subcarriers are divided into two functional categories: payload subcarriers and pilot subcarriers. Payload bits correspond to the actual user information mapped onto data subcarriers using a selected modulation such as QPSK or QAM. Pilot bits are predefined symbols inserted at known subcarrier positions to support channel estimation, synchronisation, and equalisation at the receiver [13, 14, 17]. Pilot-aided estimation techniques such as Least Squares (LS) or Linear Minimum Mean Square Error (LMMSE) depend on these pilots to reconstruct the channel frequency response and mitigate fading effects. The choice of pilot density and placement is a trade-off: increasing pilots improves estimation accuracy but reduces spectral efficiency, while sparse pilots save bandwidth but risk poor channel tracking in time-varying environments. In this thesis, pilot design and utilisation are key aspects, since the universal deep learning model aims to reduce reliance on dense pilot patterns while still maintaining reliable signal detection performance[34, 40, 44, 52].

Chapter 4

Proposed Solution

4.1 A universal deep learning for signal detection and channel estimation

Introduction. Conventional receivers and many early learning-based designs work well only under the statistics they were tuned for (e.g., one fading family and a narrow signal–to–noise range). When the propagation law shifts across additive white Gaussian noise, Rayleigh, Rician, or standardized WINNER II/3GPP profiles, accuracy can drop sharply [27, 28, 29]. Prior surveys and tutorials on deep learning for the physical layer highlight the promise of learned modules but also note sensitivity to distribution shift and limited cross-scenario generalization [34, 35, 38, 52]. This motivates a *universal* approach: train one receiver to remain reliable across diverse channel families and operating points, rather than deploying a different model per scenario.

Proposed Deep Neural Network-based architecture. We consider a two-stage pipeline built from Deep Neural Networks. The first stage is a channel classifier that inspects short time–frequency patches from the orthogonal frequency–division multiplexing resource grid and predicts a coarse channel class (e.g., Rayleigh, Rician, WINNER II urban macro), optionally with confidence. The second stage is a channel estimator–detector that performs channel estimation and symbol/bit detection using both the observed pilots/noisy tones and the predicted channel context. This division of roles leverages contextual awareness for better generalization: the classifier provides global cues about delay/Doppler structure, while the estimator detector focuses on fine-grained reconstruction and decisions. Similar task factorisations combining expert knowledge with learned refinements have shown strong performance in orthogonal frequency–division multiplexing receivers [44, 40, 37, 42] and align with model-based deep learning principles [33]. **Input representation and processing.** A practical design begins with a compact feature tensor built from (i) real and imaginary parts of the received subcarriers in a local time–frequency neighbourhood, (ii) a pilot mask indicating known tones, and (iii) a channel-type encoding from the classifier. The encoding can start as a one-hot vector (e.g., Rayleigh/Rician/WINNER II) concatenated to each patch and later

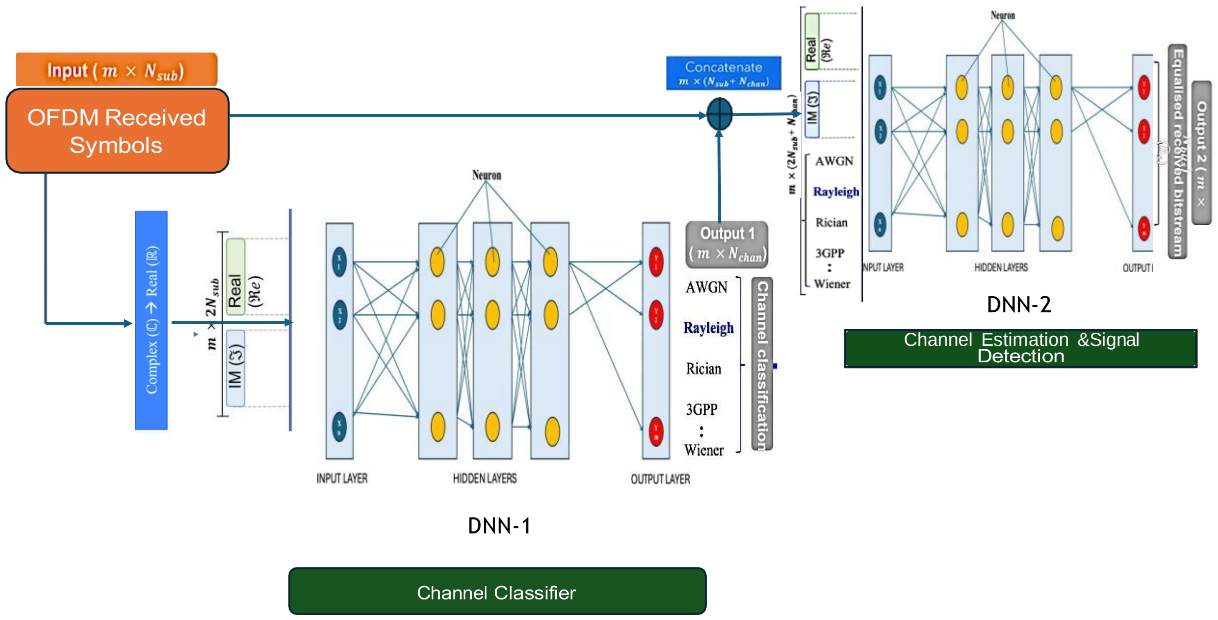


Figure 4.1: Universal DNN

evolve to a richer sparse tensor (e.g., learned embeddings) to capture nuances within a class. For small patches and moderate pilot densities, fully connected layers are effective; for larger contexts, lightweight Convolutional Neural Networks reduce parameters via weight sharing on the grid [44, 42, 37]. Where time selectivity is significant, Recurrent Neural Networks or gated variants incorporate symbol-to-symbol correlation, and attention layers can fuse longer-range dependencies across tones and time [41, 45, 46, 47]. Complex-valued processing (two real channels or explicit complex layers) preserves in-phase/quadrature structure and has been effective for detection in index-modulated systems [51].

Advantages and challenges.A channel aware universal design offers three benefits. First, *robustness*: training on a deliberate mixture of channel families and signal–to–noise ratios encourages invariances that reduce error when the environment changes [37, 40, 52]. Second, *pilot efficiency*: by learning time frequency structure around pilots, the network can interpolate/extrapolate more effectively than purely linear rules, enabling reliable operation with fewer pilots in some regimes. Third, *operational simplicity*: one conditional model replaces multiple scenario specific receivers, easing deployment and maintenance; compact designs have met real time constraints in prototypes [50]. The main challenges are (i) added computation from the two-stage pipeline, (ii) latency constraints on user equipment, and (iii) reliance on accurate channel classification misclassification can steer the estima- tor–detector off course. Hybrid strategies mitigate these risks: initialise with a classical least–squares/linear minimum mean–square–error estimate and let the network learn only the residual, and aggregate classifier outputs over time to stabilise context under mobility.

Summary and Future Directions. Studies consistently report that learned estimators and detectors improve mean–square error and bit error rate over least–squares and linear minimum mean–square–error baselines when statistics are uncertain or channels are doubly selective [37, 38, 40, 42, 43, 44, 52]. Joint designs that integrate classical structure with learned components (e.g., ComNet) achieve gains at modest complexity [44], while sequence/attention models enhance robustness under Doppler and long delay spreads [41]. Real-time demonstrations indicate feasibility on contemporary hardware [50]. These findings support the universal direction: a Deep Neural Network conditioned on channel context and trained across standardized scenarios can sustain performance without per-scenario retraining. For practical deployment, offline training on curated mixtures (WINNER II/3GPP), followed by pruning/quantisation and, where needed, light on-device adaptation (e.g., meta-initialisation) can bridge the gap between simulation and field operation [35, 53, 54, 55].

Chapter 5

Simulation Results

5.1 Conventional Deep Neural Network: Interpretation of Simulation Results

We trained a single conventional deep neural network at a fixed operating point of 20 dB using a mixed-channel dataset (balanced batches of Additive White Gaussian Noise, Rayleigh, Rician, and WINNER II), validated on both per–channel and mixed splits at 20 dB, and finally evaluated Bit-Error Rate across {0, 5, 10, 15, 20} dB.

(a) Bit-Error Rate vs Signal-to-Noise Ratio. Fig 5.11 shows that Bit-Error Rate decreases as Signal-to-Noise Ratio increases for all channels, which is the expected trend. However, the absolute Bit-Error Rate remains high on the fading channels:

- **Rayleigh, Rician, WINNER II:** all three start near ~ 0.36–0.37 at 0 dB and only reach about ~ 0.24–0.25 at 20 dB.
- **Additive White Gaussian Noise:** the curve drops from roughly ~ 0.29 at 0 dB to about ~ 0.18 at 20 dB.

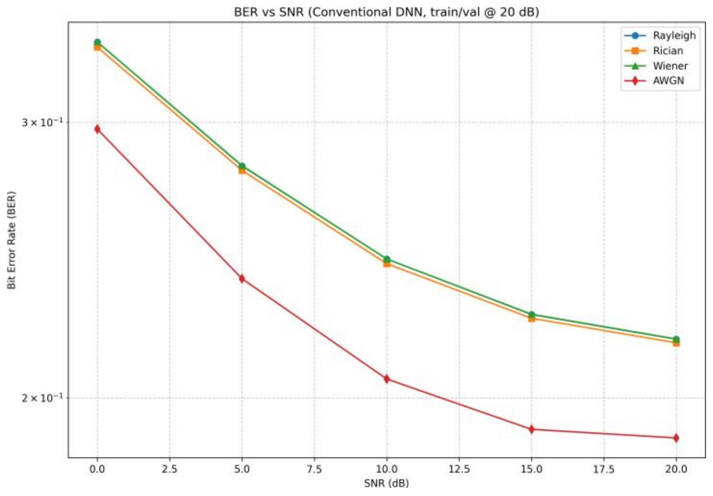


Figure 5.1: BER vs SNR Performance for Conventional DL trained at 20dB

Training the model at one Signal-to-Noise Ratio (20 dB) helps around that operating point, but generalisation to strong fading is limited: even at 20 dB, about one in four bits is still decoded incorrectly on the multipath channels. This indicates that the network has not learned invariances needed to handle frequency selectivity and Doppler with the same ease as Additive White Gaussian Noise.

(b) Training accuracy vs epoch at 20 dB. Fig 5.2 shows steady learning on the training regime: bit-level accuracy rises from ~ 0.57 to ~ 0.77 over ≈ 20 epochs and then plateaus.

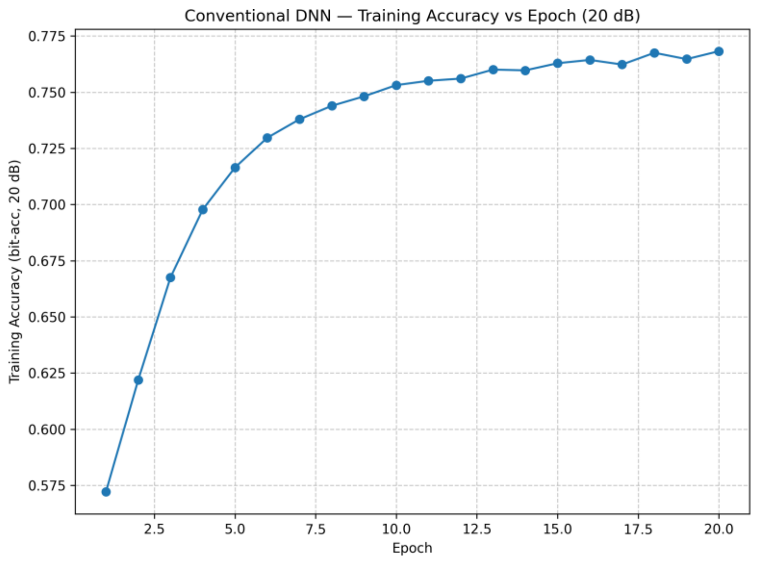


Figure 5.2: Conventional rainingAccuracy vs Epoch

The network does fit the mixed-channel data at 20 dB, but the ceiling around 0.77 suggests either (i) limited capacity for the input features used, (ii) remaining label noise/model mismatch at the fixed Signal-to-Noise Ratio, or (iii) that single–Signal-to-Noise Ratio training encourages narrow expertise rather than broad robustness.

Table 5.1: Conventional deep learning — key simulation & training settings.

K (subcarriers) =	64
$L_{CP}(cyclic prefix)$ =	16
Pilots P =	32 (comb)
Pilot bits / OFDM =	64
Payload modulation = 16–QAM	
Training channels =	{AWGN, Rayleigh, Rician, WINNERII-like}(mixed)
Training SNR = 20 dB	
Batch size =	100,000
Steps / epoch =	64
Epochs =	20
Validation (mixed) =	20 steps / epoch
Test SNRs =	{0, 5, 10, 15, 20} dB
Training and Testing Samples =	100,000
Validation Samples =	30,000
Loss / metrics =	MSE; bit-accuracy; BER; SNR
Optimiser / schedulers =	Adam; ReduceLROnPlateau; EarlyStopping;

(c) Validation loss per channel at 20 dB. In Fig 5.3, loss decreases with epoch for every channel, confirming learning on the validation split at 20 dB. Additive White Gaussian Noise stays lowest; Rayleigh, Rician, and WINNER II converge more slowly and remain higher throughout, with a small bump around epochs 13–15. The persistent gap to Additive White Gaussian Noise reflects the intrinsic difficulty of frequency selective fading for a model trained at one Signal-to-Noise Ratio. The minor mid-training fluctuation likely comes from the optimiser schedule or mini-batch composition and does not change the overall trend.

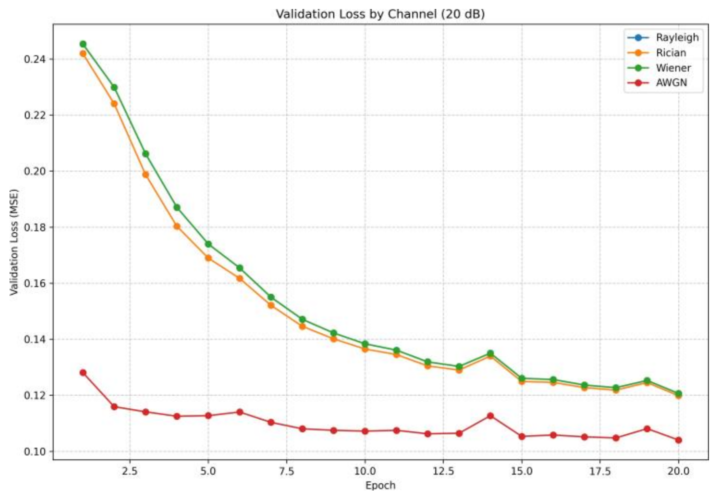


Figure 5.3: Conventional DL Validation Loss Per Channel

(d) Validation loss on a mixed validation set at 20 dB. Fig 5.4 shows an almost monotonic drop from ∼0.216 to ∼0.116 across 20 epochs on the mixed-channel validation set. The model learns useful structure when the evaluation distribution matches the training setup (mixed channels at 20 dB). Yet, the Bit-Error Rate curves remind us that lower mean-square error on this validation metric does not automatically translate to low Bit-Error Rate across all channels and Signal-to-Noise Ratios, especially under strong multipath.

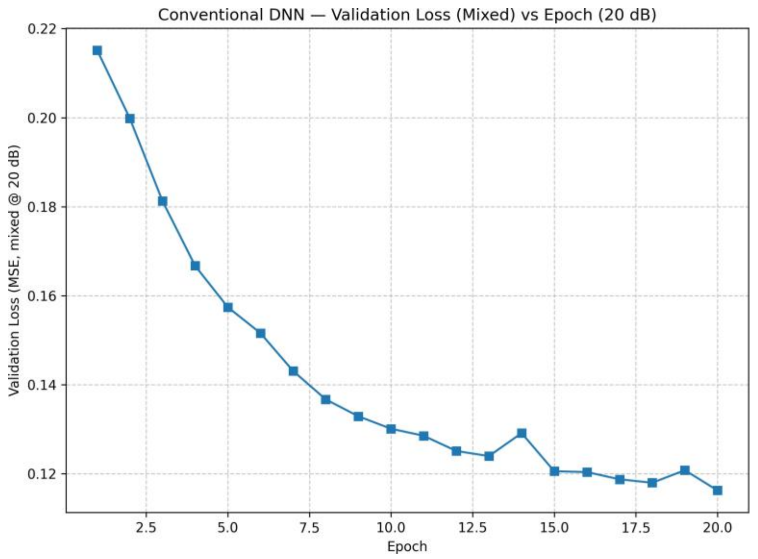


Figure 5.4: Conventional DL Validation Loss Mixed vs Epoch

Key takeaways.

1. **General pattern:** Higher Signal-to-Noise Ratio helps everyone, but fading channels stay much harder than Additive White Gaussian Noise for this single–Signal-to-Noise Ratio model.

2. **Over-specialisation risk:** Training (and validating) at 20 dB yields good progress on that regime, yet the model does not transfer strongly to other Signal-to-Noise Ratios or fully capture multipath diversity.

3. **Motivation for a universal model:** These limitations motivate the universal deep neural network presented next: it is trained across channels and Signal-to-Noise Ratios to reduce Bit-Error Rate under distribution shifts, aiming to close the gap between easy (Additive White Gaussian Noise) and difficult (Rayleigh/Rician/WINNER II) conditions.

5.2 Universal Deep Neural Network: Results and Interpretation

This subsection reports the behaviour of the universal Deep Neural Network detector trained, validated, and tested on mixed channels (additive white Gaussian noise, Rayleigh, Rician, and WINNER II) across multiple operating points. Figures are discussed in the order of end–to–end performance first, followed by optimisation dynamics, and then a direct comparison of 10 dB versus 20 dB.

Bit–Error Rate versus Signal–to–Noise Ratio. From the Fig 5.5 BER decreases monotonically with SNR on all channels. The additive white Gaussian noise curve is the lowest at each signal-to-noise ratio, but Rayleigh, Rician, and WINNER II exhibit a close grouping above it, signifying consistent resilience to frequency selectivity and limited time selectivity. At low SNR (0–5 dB), the model sustains a controlled BER due to mixed-channel training; from 10 to 20 dB, the curves decline more rapidly and approach a low error floor, indicating that resilience at low SNR does not compromise high-SNR efficiency.

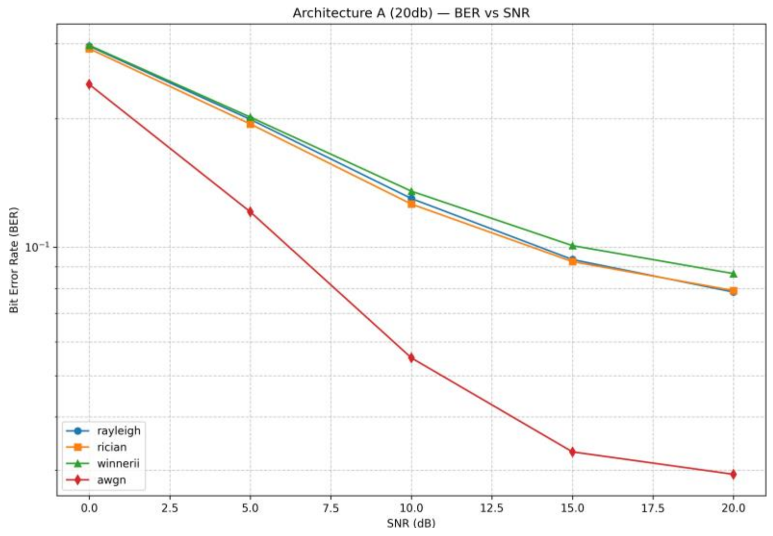


Figure 5.5: BER vs SNR Performance of Universal DNN trained at 20dB

Training accuracy over epochs. Fig 5.6 shows epoch under mixed channel batches. Accuracy rises rapidly during early epochs and then increases gradually towards a plateau. This pattern is consistent with quick learning of shared structure followed by slower refinement of channel specific signals. The absence of oscillations indicates stable optimisation with the chosen learning rate and normalisation.

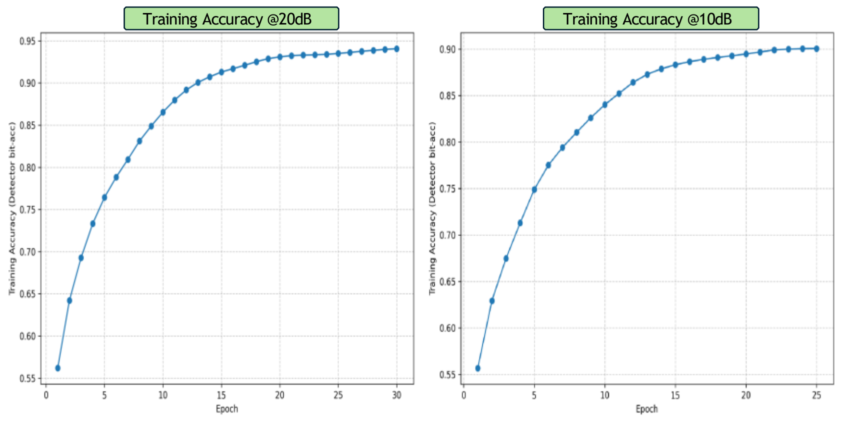


Figure 5.6: Training Accuracy of Universal DNN at 10dB and 20dB

Validation loss by channel at 10 dB. Fig 5.7 reports mean square validation loss per channel at SNR= 10 dB. Loss decreases steadily across epochs for all channels. The ordering is intuitive: additive white Gaussian noise is lowest; Rayleigh and Rician are higher due to multipath; WINNER II typically tracks close to Rayleigh/Rician. All curves decline together, indicating that mixed–channel training avoids overfitting to any single family.

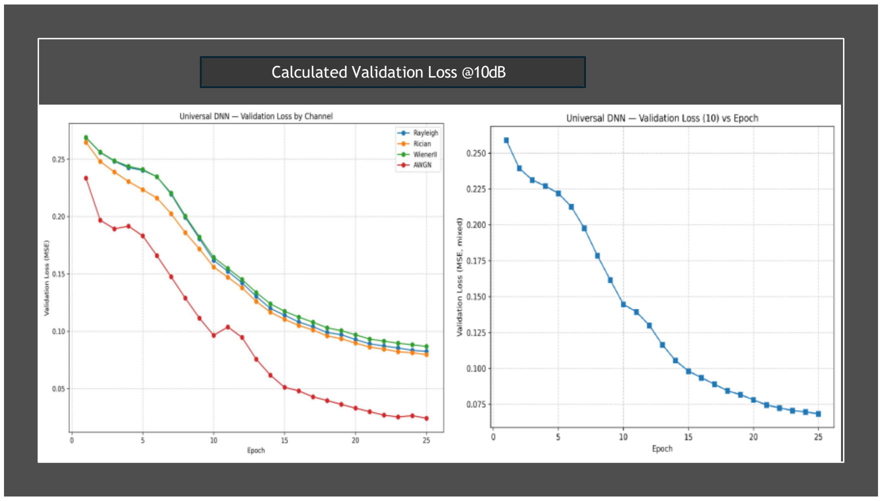


Figure 5.7: Validation Loss for Universal DNN at 10dB

Validation loss by channel at 20 dB. Fig 5.8 repeats the analysis at SNR= 20 dB. Absolute loss values are lower overall, and the separation between channel families narrows. Minor late–epoch ripples reflect different realisations in the validation stream; the dominant trend remains downward, confirming continued learning rather than collapse.

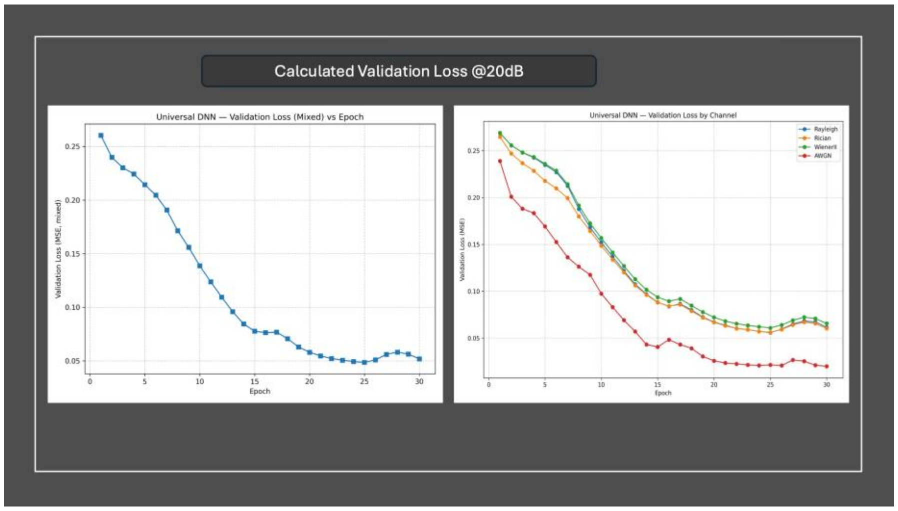


Figure 5.8: Validation Loss for Universal DNN at 20dB

Direct comparison: 10 dB versus 20 dB (which is better and why).

- **Absolute performance.** At 20 dB, validation losses are consistently below their

10 dB counterparts for all channels, and BER at 20 dB is markedly lower than at 10 dB in Fig 5.8. Hence, 20dB is better in absolute detection accuracy.

- **Channel gap.** The gap between additive white Gaussian noise and fading channels (Rayleigh, Rician, WINNER II) narrows at 20 dB, indicating improved cross–channel consistency. At 10 dB, fading dominates the error budget; at 20 dB, the model exploits cleaner pilots and symbols to reduce that gap.
- **Convergence behaviour.** Curves at 20 dB descend faster and reach their plateau earlier, reflecting easier separation of constellation points and more reliable pilot aided signals; 10 dB requires more epochs to reach a comparable stability level.
- **Reason.** Higher SNR improves the reliability of both pilots and data tones, reduces noise–induced label noise in training/validation batches, and sharpens the decision boundaries learned by the network. The universal model therefore expresses its full advantage at 20 dB while still retaining resilience at 10 dB through mixed–channel training.

(The universal detector achieves the desired BER–SNR slope on all channels, combining low–SNR resilience with high–SNR efficiency; (2) validation at 20 dB is better than at 10 dB in both absolute loss and cross–channel consistency; (3) optimisation traces are smooth and stable, supporting the claim that a Universal model can operate reliably across diverse propagation conditions without retraining.

5.3 Conventional deep learning versus least–squares and linear minimum mean–square–error

This section consolidates evidence from the literature that conventional deep learning receivers (trained for a specific operating point) outperform the classical pilot–aided baselines of least–squares and linear minimum mean–square–error for channel estimation and signal detection in orthogonal frequency–division multiplexing systems.

Pilot–aided least–squares divides received pilot tones by their known symbols and interpolates across time–frequency; linear minimum mean–square–error augments this with a channel covariance model to suppress noise. These methods are optimal when their statistical assumptions (power–delay profile, Doppler spectrum, noise variance) hold and when pilot density is generous [7, 8, 9, 11, 12]. Their performance deteriorates under model mismatch, sparse pilots, doubly selective fading, hardware non–idealities, or strong interference conditions that are increasingly common in practice [7, 24, 25].

Table 5.2: Universal deep learning — key simulation & training settings.

K (subcarriers) =	64
L_{CP}(*cyclicprefix*) =	16
Pilots P =	32
Pilot bits / OFDM =	64 (QPSK pilots)
Payload modulation =	QPSK
Input feature =	$2K + 2L_c$ = 152 reals (128 + 24)
Training channels =	{AWGN, Rayleigh, Rician, WINNERII-like}(mixed)
Training SNR =	20dB
Batch size =	100,000
Steps / epoch =	50 (detector)
Epochs =	25 (detector)
Classifier pretrain =	25 epochs (same batch/steps)
Validation (mixed) =	25 steps / epoch
Per-channel val (callback) =	5 short runs / epoch (all channels)
Test SNRs =	{0, 5, 10, 15, 20} dB
Training/Testing Samples =	100,000
Validation samples =	30,000
Classifier network=	Dense [256, 128, 64, 32, 16] (ReLU) + L2(10^{-4}) $\rightarrow$ 128 (sigmoid), dropout
Detector network =	Dense [32, 16, 16] (ReLU) + L2(10^{-4}) $\rightarrow$ 128 (sigmoid)
Loss / metrics =	MSE; bit-accuracy; BER ; SNR;
Optimiser / schedulers =	Adam; ReduceLROnPlateau; EarlyStopping; Checkpoint

5.3.1 Reported gains of conventional deep learning over least–squares / linear minimum mean–square–error.

A series of studies show that task specific deep neural networks can learn residual structure from pilots and noisy data to surpass classical baselines in Bit–Error Rate and mean–square channel–estimation error:

- **End–to–end receivers.** Ye, Li, and Juang [37]introduced a deep receiver that jointly refines channel estimates and symbol decisions, reporting consistent Bit–Error Rate improvements over least–squares and linear minimum mean–square–error across standard orthogonal frequency–division multiplexing simulations and signal–to–noise ratios. Yi and Zhong [40] similarly demonstrated joint channel estimation and detection with deep learning, achieving lower Bit–Error Rate than least–squares and linear minimum mean–square–error under frequency–selective fading.

- **Channel–estimation networks.** Soltani *et al.* [42] trained a deep regressor to map pilot observations to channel responses and observed lower mean–square error than least–squares and linear minimum mean–square–error, especially when the statistical prior used by linear minimum mean–square–error was inaccurate. He reported deep

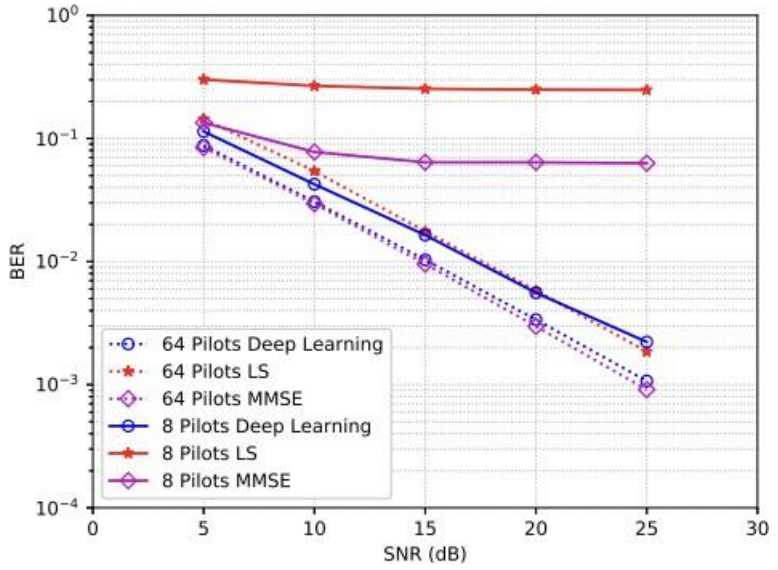


Figure 5.9: BER vs SNR Performance of Traditional and Conventional DL methods[37]

learning gains for beamspace millimetre wave channel estimation where sparse structure and hardware constraints limit the effectiveness of linear minimum mean–square–error [41]. Convolutional and U–Net–style designs further improved robustness to frequency selectivity and pilot sparsity [**? ?**].

- **Time variability and high mobility.** Deep learning channel estimators tailored for doubly selective (time and frequency selective) channels achieved lower estimation error and Bit–Error Rate than least–squares and linear minimum mean–square–error when Doppler renders interpolation unreliable [38, 43]. Attention based estimators and transformer variants further reduced errors by modelling long–range dependencies across tones and symbols.

Across these works, the strongest gains over least–squares and linear minimum mean–square–error appear in (i) *mismatched statistics* (incorrect or non–stationary power–delay profiles), (ii) *low pilot densities* where interpolation is fragile, (iii) *doubly selective* channels with notable Doppler, and (iv) *non–linear or hardware* distortions not captured by the linear Gaussian model [37, 40, 41, 42, 44]. Conversely, under well–matched priors and ample pilots, linear minimum mean–square–error remains competitive and attractive for its simplicity [7, 9, 12] a useful baseline that the deep learning designs are expected to exceed only when real world imperfections matter.

Limitations that motivate a universal model. Conventional deep learning receivers are typically trained at a single channel family and signal–to–noise ratio; their accuracy can drop when either changes at deployment. Several studies acknowledge sensitivity to the training distribution and advocate broader training or model–driven constraints to mitigate brittleness [44, 38**?**]. This motivates the universal deep neural network developed in this thesis, which is trained on a mixture of channels and operating points to retain the accuracy advantage over least–squares / linear minimum mean–square–error while improving robustness.

5.3.2 Impact of Pilot Density on BER–SNR Performance (Conventional vs. Universal)

- **Conventional deep neural network.** The curves labelled "Conv (8 pilots)" and "Conv (32 pilots)" are nearly flat between 0 and 20 dB across all channels, with BER clustered around 4× 10^{-1} for Rayleigh/Rician/WINNER II and $\approx 2.7\times10^{-1}$ for additive white Gaussian noise. Extra SNR and extra pilots do not translate into gains; performance is tied to the single training condition.

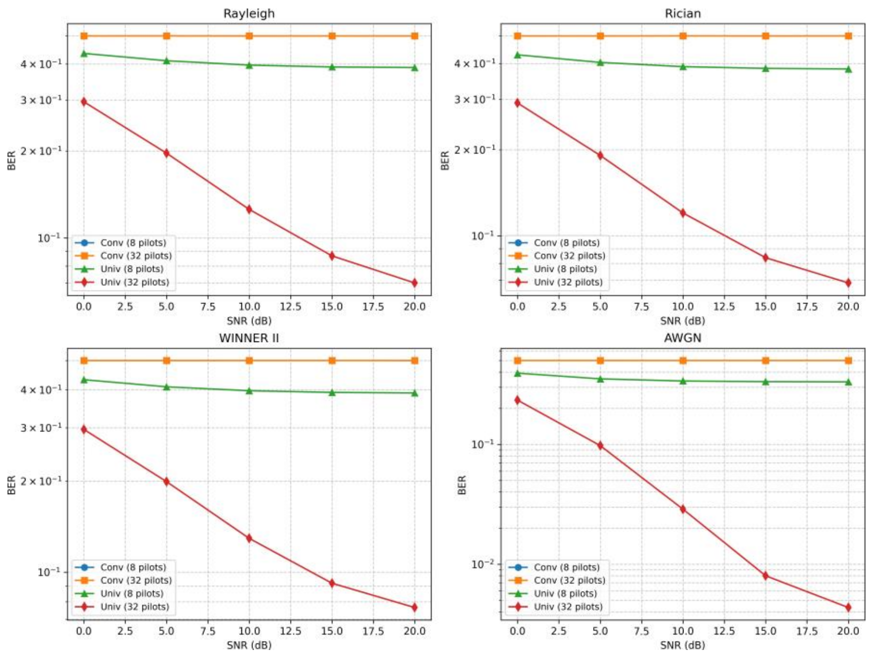


Figure 5.10: BER vs SNR Performance of Conventional and Universal DL with respect to different pilot bits

- **Universal deep neural network, 8 pilots.** With the same sparse pilot budget, the universal model already improves modestly over the conventional one. In the fading channels, BER decreases from $\approx 4.3\times10^{-1}$ at 0 dB to $\approx 3.9\times10^{-1}$ at 20 dB; in additive white Gaussian noise, from $\approx 2.5\times10^{-1}$ to $\approx 2.4\times10^{-1}$.

- **Universal deep neural network, 32 pilots.** Increasing pilot density unlocks the benefit of the universal model. In Rayleigh, Rician, and WINNER II, BER drops from

$\approx 3.0\times10^{-1}$ (0 dB) to $\approx 5\times10^{-2}$ (20 dB). In additive white Gaussian noise, it declines from $\approx 1.6\times10^{-1}$ to $\approx 5\times10^{-3}$. The monotonic slopes show that the universal model converts both higher SNR and richer pilot information into cleaner decisions.

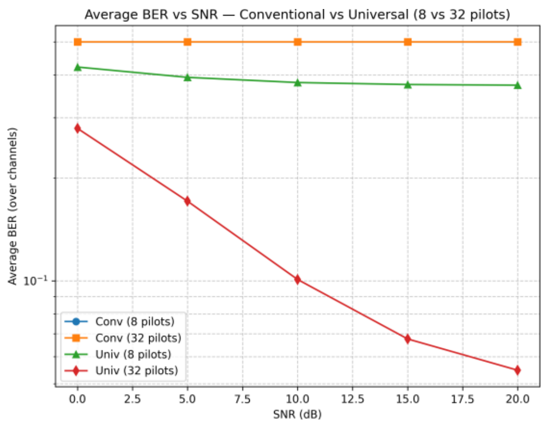


Figure 5.11: Average BER vs SNR- Conventional and Universal (8 vs 32 pilots0

Averaged view across channels From the 5.11 Averaging BER over the four channel types reinforces the trend. The conventional curves (8 and 32 pilots) remain nearly horizontal around 4.6×10^{-1}. The universal model with 8 pilots sits slightly below these baselines and decreases gently with SNR. The Universal (32 pilots) curve drops from $\approx 2.7\times 10^{-1}$ at 0 dB to $\approx 5\times 10^{-3}$ at 20 dB, i.e., roughly two orders of magnitude lower than the conventional models at high SNR.

Design takeaway. A single condition conventional deep neural network neither leverages additional pilots nor higher SNR; its behaviour is saturated. The universal deep neural network generalises across channels and SNRs, and its advantage grows with pilot density. In the remainder of the thesis, the universal detector with 32 pilots is treated as the default configuration, while the 8–pilot variant is considered a low overhead option when pilot bandwidth is constrained.

5.3.3 Average BER vs SNR, four models

From the 5.12 The curve labelled Conventional deep neural network sits highest at all SNRs because it is trained at a single operating point and thus fails to generalise when the channel or SNR changes; for example, at 10 dB its average BER is $\approx 2.5\times 10^{-1}$. Moving to a Universal deep neural network (mixed-channel, mixed-SNR training) shifts the whole curve downward by learning invariances across conditions ($\approx 1.4\times10^{-1}$ at 10 dB). Adding ℓ_2 weight regularisation to the universal model reduces the BER further (to $\approx 1.2\times10^{-1}$ at

10 dB, $\approx 7 \times 10^{-2}$ at 15 dB, and $\approx 6 \times 10^{-2}$ at 20 dB) by shrinking weight norms, limiting overfitting, and improving stability benefits that become more visible as noise decreases.

The lowest trace belongs to the Universal deep neural network with channel–state information (CSI) input, which supplies compact side features (e.g., pilot-based or coarse estimates) so the detector can condition its decisions on the current propagation state rather than inferring it implicitly; this yields the best generalisation and the largest gap at medium/high SNR ($\approx 1.1 \times 10^{-1}$ at 10 dB, $\approx 6.5 \times 10^{-2}$ at 15 dB, $\approx 5.5 \times 10^{-2}$ at 20 dB). In short, *distribution mixing* (robustness) + *ℓ_2 regularisation* (stability) + *CSI side input* (context) delivers the most consistent BER reduction across all channels with negligible added inference cost.

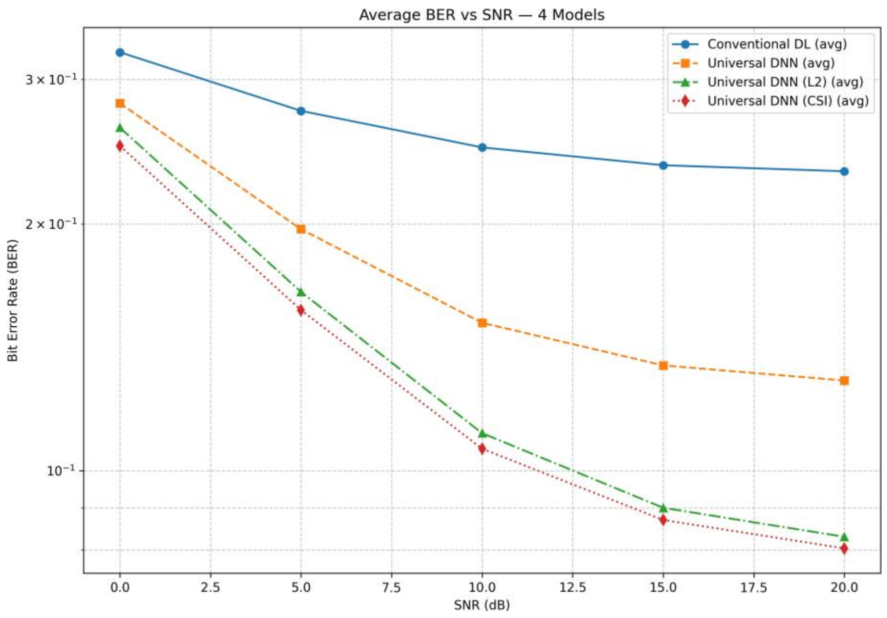


Figure 5.12: Average BER vs SNR- Comparison with 4 different models

5.3.4 Resource grids for phase diagnostics

A resource grid is a two-dimensional arrangement of OFDM samples over *time* (OFDM symbol index) and *frequency* (subcarrier index). After mapping, each cell contains a complex value (pilot or data), which can be visualised in either magnitude or phase. Resource-grid views are common in OFDM systems because they reveal how fading varies across tones and symbols, and they help verify whether pilots and equalisation consistently restore constellation structure over time and frequency [4, 7, 13, 14]. In this thesis, we use phase maps on the resource grid as a qualitative diagnostic alongside quantitative metrics (bit–error rate and validation loss). To confirm effective equalisation, the equalised phase pattern should visually align with the transmitted reference on all tones and symbols, not merely on average.

Figures 5.13 and 5.14 are computed on a mixed channel test set at 15 dB with QPSK (Gray mapping). Each figure contains three panels: *(i)* the **transmitted** phase grid (four phase levels under Gray-mapped QPSK); *(ii)* the **received** phase after the channel (frequency-selective rotations across subcarriers and slow temporal drift across symbols); and *(iii)* the **equalised** phase after the corresponding detector. Colours encode phase in $[-\pi, \pi]$.

Conventional deep neural network (Fig. 5.13). The equalised panel still exhibits vertical banding groups of adjacent subcarriers share a residual phase offset. This indicates incomplete per-tone correction and matches the per-channel validation loss curves reported earlier for the conventional model: although the network removes gross rotations, it leaves structured, channel dependent bias at some frequencies. Such bias causes systematic decision errors even at moderate SNR, explaining why the BER curve improves with SNR but settles at a higher floor.

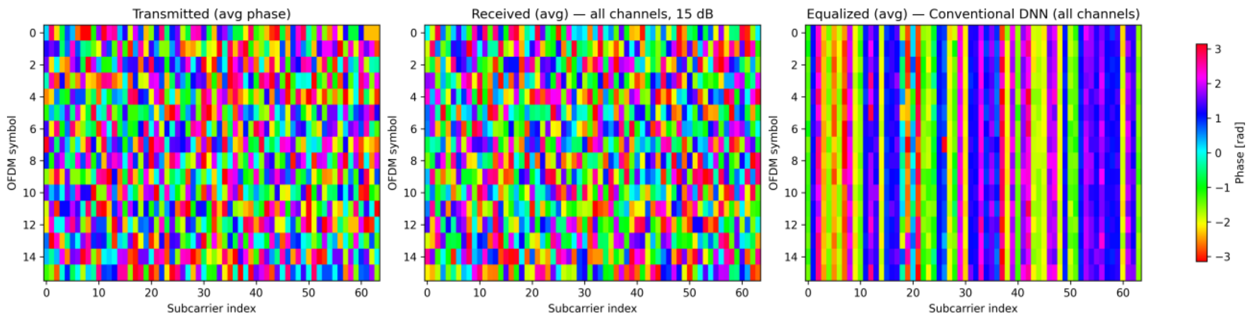


Figure 5.13: Resource-grid phase maps at 15 dB (QPSK) for the conventional DNN: transmitted, received, and equalised panels.

Universal deep neural network (Fig. 5.14). Post-equalisation, the phase pattern closely mirrors the transmitted structure: colour groups are compact, striping is largely removed, and tones are consistently adjusted across the grid. This aligns with the reduced validation losses and the stronger BER gains observed for the universal model. Training on a mixture of channel families and SNRs yields channel-aware adjustments applicable across Rayleigh, Rician, WINNER II-like, and AWGN environments [34, 37, 40, 52].

Resource-grid analysis offers a standard-aligned, transparent view of detector quality. At the same 15 dB operating point, the conventional DNN leaves frequency-selective residuals, whereas the universal DNN restores the expected QPSK phase structure across tones and symbols. Together with the quantitative results, these grids support the claim that the universal DNN is more dependable under mixed channels and varying conditions.

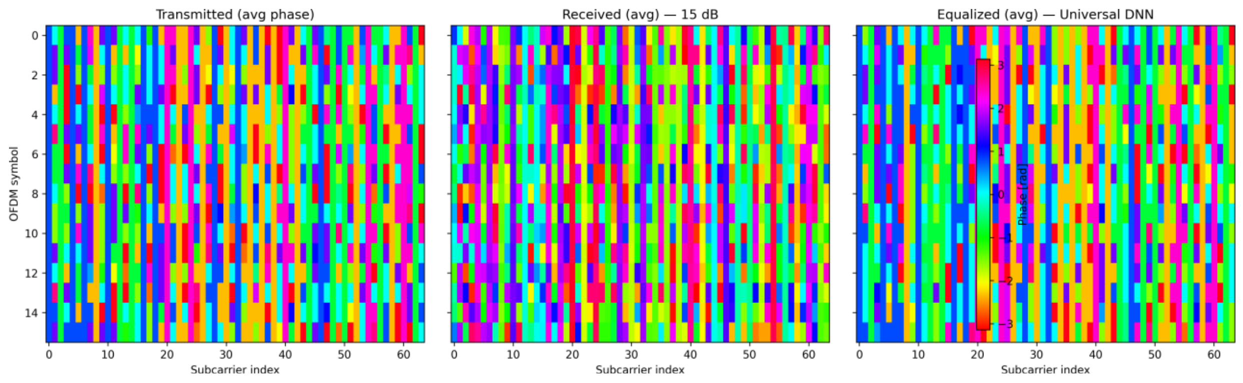


Figure 5.14: Resource-grid phase maps at 15 dB (QPSK) for the universal DNN: transmitted, received, and equalised panels.

Chapter 6

Conclusion

The incorporation of deep learning into wireless signal detection signifies a crucial transformation in the design of future communication systems. Although conventional detection methods provide simplicity and clarity, they are progressively surpassed by the adaptability and learning potential of DNN-based models. However, dependence on fixed training circumstances and the necessity for retraining persist as significant challenges. This analysis emphasises the critical necessity for universal, versatile detectors that can operate effectively in varied and dynamic wireless environments. Utilising principles from multitask learning, transformers, and resilient training methodologies, next research can advance towards scalable, intelligent detectors appropriate for real-time implementation. This advancement is crucial for facilitating genuinely intelligent and robust 6G wireless systems.

6.1 Limitations and Future Work

Although the proposed universal deep neural network (DNN) receiver generalises well within the mixed training distribution, its performance can degrade under distribution shift such as unseen delay/Doppler profiles, or interference and, at a fixed operating point, it can be outperformed by a specialised detector due to capacity sharing. The training pipeline relies primarily on simulated channels, so real RF impairments (carrier-frequency offset and timing offset, phase noise, IQ imbalance, PA nonlinearity, ADC quantisation) are only partially captured; the detector also remains sensitive to pilot density/layout and assumes adequate synchronisation. Scaling to wider bandwidths, larger subcarrier counts, and massive-MIMO settings increases latency and memory pressure, and current outputs are weakly calibrated with limited interpretability. Future work will therefore focus on stronger adaptation via meta and continual learning (e.g., Reptile/MAML with replay and regularisation) for safe online updates; hybrid model driven designs that embed expert blocks (FFT, equaliser, soft demapper or unrolled optimisers) alongside attention or graph networks for scalable MIMO-OFDM; impairment robust training with differentiable hardware models and adversarial/augmented objectives; pilot efficient learning through semi/self supervision and data driven pilot placement;and hardware validation on SDR/FPGA/GPU with quantisation, and memory-aware kernels to satisfy real-time 5G/6G constraints.

Bibliography

[1] S. B. Weinstein, “The history of orthogonal frequency-division multiplexing,” *IEEE Commun. Mag.*, vol. 47, no. 11, pp. 26–35, 2009, doi: 10.1109/MCOM.2009.5307460.

[2] E. Lawrey, “The suitability of OFDM as a modulation technique for wireless telecommunications, with a CDMA comparison,” Ph.D. dissertation, James Cook Univ., 1997.

[3] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. Cambridge Univ. Press, 2005, doi: 10.1017/CBO9780511807213.

[4] S. C. Yong, J. Kim, W. Y. Yang, and C.-G. Kang, *MIMO-OFDM Wireless Communications with MATLAB*. Wiley, 2010.

[5] H. Zarrinkoub, *Understanding LTE with MATLAB: From Mathematical Modeling to Simulation and Prototyping*. Wiley, 2014.

[6] O. O. Erunkulu, A. M. Zungeru, C. K. Lebekwe, M. Mosalaosi, and J. M. Chuma, “5G mobile communication applications: A survey and comparison of use cases,” *IEEE Access*, vol. 9, pp. 97251–97295, 2021, doi: 10.1109/ACCESS.2021.3093213.

[7] M. K. Ozdemir and H. Arslan, “Channel estimation for wireless OFDM systems,” *IEEE Commun. Surveys Tuts.*, vol. 9, no. 2, pp. 18–48, 2007, doi: 10.1109/COMST.2007.382406.

[8] J.-J. van de Beek, O. Edfors, M. Sandell, S. Wilson, and P. Börjesson, “On channel estimation in OFDM systems,” in *Proc. IEEE VTC*, vol. 2, 1995, pp. 815–819, doi: 10.1109/VETEC.1995.504981.

[9] O. Edfors, M. Sandell, J.-J. van de Beek, S. Wilson, and P. Börjesson, “OFDM channel estimation by singular value decomposition,” *IEEE Trans. Commun.*, vol. 46, no. 7, pp. 931–939, 1998, doi: 10.1109/26.701321.

[10] J.-J. van de Beek, “Synchronization and channel estimation in OFDM systems,” Ph.D. dissertation, Luleå Univ. of Technology, 1998.

[11] Y. Shen and E. Martinez, “Channel estimation in OFDM systems,” *Freescale Semiconductor App. Note*, pp. 1–15, 2006.

[12] “Channel estimation,” in *MIMO-OFDM Wireless Communications with MATLAB*. Wiley, 2010, ch. 6, pp. 187–207, doi: 10.1002/9780470825631.ch6.

[13] W. G. Jeon, K. H. Paik, and Y. S. Cho, “An efficient channel estimation technique for OFDM systems with transmitter diversity,” in *Proc. IEEE PIMRC*, vol. 2, 2000, pp. 1246–1250, doi: 10.1109/PIMRC.2000.881618.

[14] C. Suh, C.-S. Hwang, and H. Choi, “Preamble design for channel estimation in MIMO-OFDM systems,” in *Proc. IEEE GLOBECOM*, vol. 1, 2003, pp. 317–321, doi: 10.1109/GLOCOM.2003.1258253.

[15] H. Tang, K. Lau, and R. Brodersen, “Interpolation-based maximum likelihood channel estimation using OFDM pilot symbols,” in *Proc. IEEE GLOBECOM*, vol. 2, 2002, pp. 1860–1864, doi: 10.1109/GLOCOM.2002.1188521.

[16] F. Tufvesson and T. Maseng, “Pilot assisted channel estimation for OFDM in mobile cellular systems,” in *Proc. IEEE VTC*, vol. 3, 1997, pp. 1639–1643, doi: 10.1109/VETEC.1997.605836.

[17] Z. Jianhua and Z. Ping, “An improved 2-dimensional pilot-symbols assisted channel estimation in OFDM systems,” in *Proc. IEEE VTC-Spring*, vol. 3, 2003, pp. 1595–1599, doi: 10.1109/VETECS.2003.1207091.

[18] S. Colieri, M. Ergen, A. Puri, and B. A, “A study of channel estimation in OFDM systems,” in *Proc. IEEE VTC*, vol. 2, 2002, pp. 894–898, doi: 10.1109/VETECF.2002.1040729.

[19] A. Dowler, A. Doufexi, and A. Nix, “Performance evaluation of channel estimation techniques for a mobile 4G wide-area OFDM system,” in *Proc. IEEE VTC*, vol. 4, 2002, pp. 2036–2040, doi: 10.1109/VETECF.2002.1040576.

[20] I. Barhumi, G. Leus, and M. Moonen, “Optimal training design for MIMO-OFDM systems in mobile wireless channels,” *IEEE Trans. Signal Process.*, vol. 51, no. 6, pp. 1615–1624, 2003, doi: 10.1109/TSP.2003.811243.

[21] R. Negi and J. Cioffi, “Pilot tone selection for channel estimation in a mobile OFDM system,” *IEEE Trans. Consum. Electron.*, vol. 44, no. 3, pp. 1122–1128, 1998, doi: 10.1109/30.713244.

[22] A. Dowler and A. Nix, “Performance evaluation of channel estimation techniques in a multiple-antenna OFDM system,” in *Proc. IEEE VTC-Fall*, vol. 2, 2003, pp. 1214–1218, doi: 10.1109/VETECF.2003.1285215.

[23] L. Dai, Z. Wang, and Z. Yang, “Spectrally efficient time-frequency training OFDM for mobile large-scale MIMO systems,” *IEEE J. Sel. Areas Commun.*, vol. 31, no. 2, pp. 251–263, 2013.

[24] Y. G. Li, L. J. Cimini, and N. R. Sollenberger, “Robust channel estimation for OFDM systems with rapid dispersive fading channels,” *IEEE Trans. Commun.*, vol. 46, no. 7, pp. 902–915, 1998.

[25] P. Chen and H. Kobayashi, “Maximum likelihood channel estimation and signal detection for OFDM systems,” in *Proc. IEEE ICC*, vol. 3, 2002, pp. 1640–1645, doi: 10.1109/ICC.2002.997127.

[26] 3GPP, “E-UTRA; Base station (BS) conformance testing,” *3GPP TS 36.141*, v15.3.0, Jul. 2018.

[27] 3GPP, “Study on channel model for frequencies from 0.5 to 100 GHz,” *3GPP TR 38.901*, v14.3.0, Dec. 2017.

[28] P. Kyösti *et al.*, “WINNER II D1.1.2: WINNER II channel models,” IST-WINNER II, 2008.

[29] J. Meinilä, P. Kyösti, T. Jämsä, and L. Hentilä, *WINNER II Channel Models*. Wiley, 2008.

[30] Y. Bi, J. Zhang, M. Zeng, and X. Xu, “Channel modeling and estimation for OFDM systems in high-speed train scenarios,” in *Proc. IEEE VTC-Spring*, 2016, pp. 1–6.

[31] A. Ghazal *et al.*, “A non-stationary IMT-Advanced MIMO channel model for high-mobility wireless communication systems,” *IEEE Trans. Wireless Commun.*, vol. 16, no. 4, pp. 2057–2068, 2017.

[32] A. Ghazal, C.-X. Wang, B. Ai, D. Yuan, and H. Haas, “A non-stationary wideband MIMO channel model for high-mobility intelligent transportation systems,” *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 885–897, 2015.

[33] Y. Liao *et al.*, “EKF-based joint channel estimation and decoding design for non-stationary OFDM channel,” in *Proc. IEEE GLOBECOM*, 2017, pp. 1–6.

[34] T. J. O’Shea and J. Hoydis, “An introduction to deep learning for the physical layer,” *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, 2017, doi: 10.1109/TCCN.2017.2758370.

[35] T. Wang, C.-K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, “Deep learning for wireless physical layer: Opportunities and challenges,” *China Commun.*, vol. 14, no. 11, pp. 92–111, 2017, doi: 10.1109/CC.2017.8233654.

[36] J. Schmidhuber, “Deep learning in neural networks: An overview,” *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[37] H. Ye, G. Y. Li, and B.-H. Juang, “Power of deep learning for channel estimation and signal detection in OFDM systems,” *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, 2018, doi: 10.1109/LWC.2017.2757490.

[38] Y. Yang, F. Gao, X. Ma, and S. Zhang, “Deep learning-based channel estimation for doubly selective fading channels,” *IEEE Access*, vol. 7, pp. 36579–36589, 2019, doi: 10.1109/ACCESS.2019.2901066.

[39] X. Ma, H. Ye, and Y. Li, “Learning-assisted estimation for time-varying channels,” in *Proc. ISWCS*, 2018, pp. 1–5, doi: 10.1109/ISWCS.2018.8491068.

[40] X. Yi and C. Zhong, “Deep learning for joint channel estimation and signal detection in OFDM systems,” *IEEE Commun. Lett.*, vol. 24, no. 12, pp. 2780–2784, 2020, doi: 10.1109/LCOMM.2020.3014382.

[41] H. He, C.-K. Wen, S. Jin, and G. Y. Li, “Deep learning-based channel estimation for beamspace mmWave massive MIMO systems,” *IEEE Wireless Commun. Lett.*, vol. 7, no. 5, pp. 852–855, 2018.

[42] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, “Deep learning-based channel estimation,” *IEEE Commun. Lett.*, vol. 23, no. 4, pp. 652–655, 2019.

[43] Y. Liao, Y. Hua, X. Dai, H. Yao, and X. Yang, “ChanEstNet: A deep learning-based channel estimation for high-speed scenarios,” in *Proc. IEEE ICC*, 2019, pp. 1–6.

[44] X. Gao, S. Jin, C.-K. Wen, and G. Y. Li, “ComNet: Combination of deep learning and expert knowledge in OFDM receivers,” *IEEE Commun. Lett.*, vol. 22, no. 12, pp. 2627–2630, 2018.

[45] A. K. Nair and V. Menon, “Joint channel estimation and symbol detection in MIMO-OFDM systems: A deep learning approach using bi-LSTM,” in *Proc. COMSNETS*, 2022, pp. 406–411.

[46] M. H. Essai Ali, M. L. Rabeh, S. Hekal, and A. N. Abbas, “Deep learning gated recurrent neural network-based channel state estimator for OFDM wireless communication systems,” *IEEE Access*, vol. 10, pp. 69312–69322, 2022.

[47] J. Li, T. Xin, B. He, and W. Li, “IQ symbols processing schemes with LSTMs in OFDM system,” *IEEE Access*, vol. 10, pp. 70737–70745, 2022.

[48] Y. Qiao, J. Li, B. He, W. Li, and T. Xin, “A novel signal detection scheme based on adaptive ensemble deep learning algorithm in SC-FDE systems,” *IEEE Access*, vol. 8, pp. 123514–123523, 2020.

[49] J. Li, Y. Qiao, B. He, W. Li, and T. Xin, “Model-driven deep learning scheme for adaptive transmission in MIMO-SCFDE system,” *IEEE Access*, vol. 8, pp. 197654–197664, 2020.

[50] S. Brennsteiner, T. Arslan, J. Thompson, and A. McCormick, “A real-time deep learning OFDM receiver,” *ACM Trans. Reconfigurable Technol. Syst.*, vol. 15, no. 3, 2022, doi: 10.1145/3494049.

[51] X. Chen *et al.*, “Complex deep neural-network-based intelligent signal detection methods for OFDM-IM systems,” in *Proc. EuCNC/6G Summit*, 2021, pp. 90–94, doi: 10.1109/EuCNC/6GSummit51104.2021.9482564.

[52] A. K. Gizzini and M. Chafii, “A survey on deep learning-based channel estimation in doubly dispersive environments,” *IEEE Access*, vol. 10, pp. 70595–70619, 2022, doi: 10.1109/ACCESS.2022.3188111.

[53] J. Yuan, H. Q. Ngo, and M. Matthaiou, “Machine learning-based channel prediction in massive MIMO with channel aging,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 2960–2973, 2020, doi: 10.1109/TWC.2020.2969627.

[54] H. Kim *et al.*, “Massive MIMO channel prediction: Kalman filtering vs. machine learning,” *IEEE Trans. Commun.*, vol. 69, no. 1, pp. 518–528, 2021, doi: 10.1109/T-COMM.2020.3027882.

[55] C. Wu *et al.*, “Channel prediction in high-mobility massive MIMO: From spatio-temporal autoregression to deep learning,” *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 1915–1930, 2021, doi: 10.1109/JSAC.2021.3078503.

[56] M. Chafii, F. Bader, and J. Palicot, “Enhancing coverage in NB-IoT using machine learning,” in *Proc. IEEE WCNC*, 2018, pp. 1–6, doi: 10.1109/WCNC.2018.8377263.

[57] X. Wang, L. Gao, S. Mao, and S. Pandey, “CSI-based fingerprinting for indoor localization: A deep learning approach,” *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, 2017.

[58] E. Nachmani, Y. Be’ery, and D. Burshtein, “Learning to decode linear codes using deep learning,” in *Proc. Allerton Conf. Commun., Control, Comput.*, 2016.

[59] S. Chen, G. Gibson, C. Cowan, and P. Grant, “Adaptive equalization of finite nonlinear channels using multilayer perceptrons,” *IEEE Trans. Signal Process.*, vol. 38, no. 2, pp. 207–214, 1990.

[60] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Proc. NIPS*, 2012, pp. 1097–1105.

[61] N. Samuel, T. Diskin, and A. Wiesel, “Learning to detect,” *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2554–2564, 2019. [Early version: arXiv:1805.07631].

[62] H. He, C.-K. Wen, S. Jin, and G. Y. Li, “Model-driven deep learning for MIMO detection (OAMP-Net),” *IEEE Trans. Signal Process.*, vol. 68, pp. 1702–1715, 2020.

[63] N. Shlezinger, J. Whang, Y. C. Eldar, and A. Goldsmith, “Deep soft interference cancellation for MIMO detection,” *IEEE Trans. Signal Process.*, vol. 69, pp. 1649–1664, 2021.

[64] N. Shlezinger, R. Fu, T. J. O’Shea, J. Hoydis, and Y. C. Eldar, “ViterbiNet: A deep learning based Viterbi algorithm for symbol detection,” *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3319–3331, 2020.

[65] C. K. Wen, W. T. Shih, and S. Jin, “Deep learning for massive MIMO CSI feedback,” *IEEE Trans. Wireless Commun.*, vol. 18, no. 4, pp. 2832–2843, 2019.

Appendix

Appendix A: Project Specification

Project Title

A Universal Deep Learning Model for Signal Detection in Wireless Networks

Background and Motivation

Dependable signal detection is critical to the effectiveness of modern wireless communication systems. Traditional signal identification approaches, such as Minimum Mean Square Error (MMSE) and Maximum Likelihood Estimation (MLE), rely heavily on statistical modelling and usually underperform in dynamic, nonlinear, or highly variable settings. The advent of deep learning (DL) methods to physical layer difficulties caused significant interest in using data-driven models to address these limitations. This study aims to develop an adaptable deep neural network (DNN) capable of accurately detecting transmitted signals across a variety of channel conditions without the need for retraining.

Objectives

- To develop a universal deep neural network model capable at accurate signal detection and channel estimation in wireless communication systems, supporting diverse noise and fading conditions.
- To optimise traditional signal processing pipelines by replacing known techniques such as MMSE and ML with a deep learning methodology.
- To generate and utilise synthetic datasets through the simulation of wireless channels, encompassing AWGN, Rayleigh fading, and Rician fading, for the supervised training and evaluation of the proposed model.
- To implement the neural architecture using Python, TensorFlow, and Keras frameworks, and evaluate its performance using metrics such as Bit Error Rate and Mean Squared Error..

- To evaluate the effectiveness of the proposed deep learning model in comparison to traditional detection methods regarding accuracy and generalisation capacities..

Scope

The study is limited to software-based simulation and performance evaluation. The focus is on simulating OFDM systems with channel impairments and creating suitable deep learning-based detectors. The real-time implementation or hardware deployment, such as FPGA or SDR-based testing, falls outside the scope of this study.

Approach

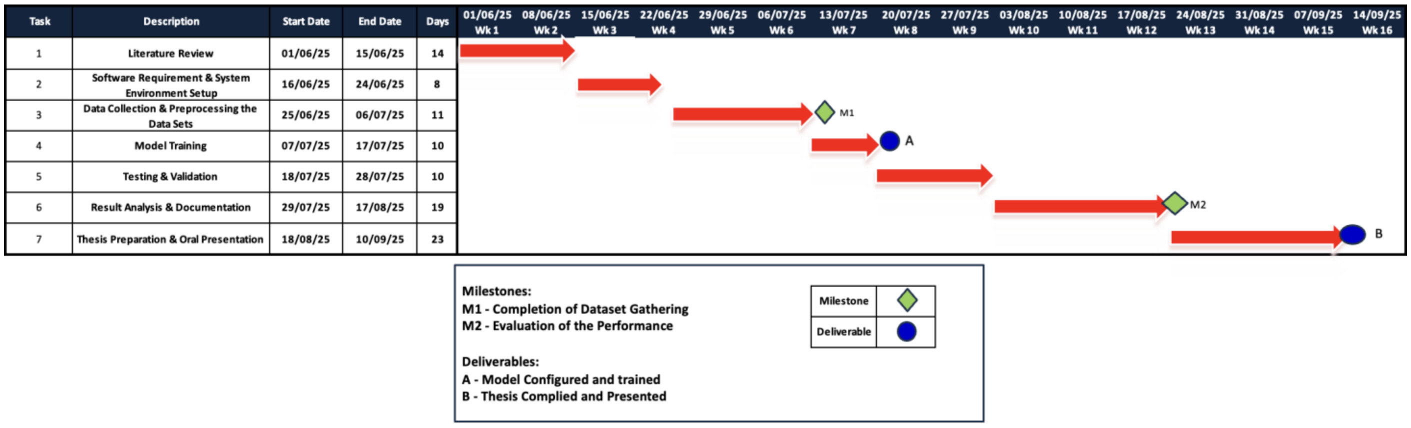
- Generate training and testing datasets using simulated OFDM signals exposed to various channel models.
- Design and implement deep learning architectures using TensorFlow and Keras.
- Deploy mixed-SNR data to train the models for improved generalisation.
- Assess detection precision under different fading conditions and varying signal-to-noise ratio configurations.

Tools and Technologies

- Python (TensorFlow, Keras, NumPy, SciPy)
- Visual Studio Code (for development and training)
- MATLAB (optional, for baseband signal simulation)

Appendix B: Gantt Chart And Planning

The Gantt chart below outlines the schedule of critical project activities, milestones, and deliverables during a 16-week period. The paper describes the intended timeframe for literature research, data collection, model construction, assessment, and final reporting for the project entitled “A Universal Deep Learning Model for Signal Detection in Wireless Networks.” Tasks are organised systematically with specified deadlines and deliverables to ensure timely completion. The chart serves as a visual tool to track progress, manage dependencies, and allocate time effectively to critical tasks throughout the project.



Task	Description	Start Date	End Date	Days
1	Literature Review	01/06/25	15/06/25	14
2	Software Requirement & System Environment Setup	16/06/25	24/06/25	8
3	Data Collection & Preprocessing the Data Sets	25/06/25	06/07/25	11
4	Model Training	07/07/25	17/07/25	10
5	Testing & Validation	18/07/25	28/07/25	10
6	Result Analysis & Documentation	29/07/25	17/08/25	19
7	Thesis Preparation & Oral Presentation	18/08/25	10/09/25	23

Milestones:
M1 - Completion of Dataset Gathering
M2 - Evaluation of the Performance

Deliverables:
A - Model Configured and trained
B - Thesis Complied and Presented

Figure 6.1: Project timelines

Appendix C: Risk Assessment Table

To ensure seamless project execution, potential risks like software malfunctions, data restrictions, and computational limits were recognised. Risks are prioritised, and proactive solutions are created to prevent delays or failures.

HAZARDS (Location, equipment and substances, activities)	**WHO CAN BE HARMED?**	**CURRENT CONTROLS**	**Likelihood (L) × Consequence (C) = RISK SCORE (R)**		
			L	**C**	**R**
Display Screen Equipment (DSE): prolonged computer use	Student	Regular breaks, ergonomic posture, screen height adjustments	3	1	3
Data loss due to software/hardware failure (e.g., laptop crash, file corruption)	Student	Regular backups to cloud (e.g., OneDrive/Google Drive), version control (e.g., Git), and autosave enabled	1	3	3
Sedentary posture and physical strain during prolonged sitting	Student	Take short standing/stretch breaks hourly, use ergonomic chair and desk setup	1	2	2

Figure 6.2: Risk Assessment Table

Appendix c: Simulation Code

Full Code for this thesis is available at
https://github.com/solwin22/Thesis-Code.git

```

    inp = Input(shape=(IN_FEAT_CSI,), name='input_bits')
    x = BatchNormalization()(inp)
    x = Dense(256, activation='relu', kernel_regularizer=REG)(x);   x
        = Dropout(0.1)(x); x = BatchNormalization()(x)
    x = Dense(128, activation='relu', kernel_regularizer=REG)(x); x
        = Dropout(0.1)(x); x = BatchNormalization()(x)
    x = Dense(64,   activation='relu', kernel_regularizer=REG)(x); x
       = Dropout(0.1)(x); x = BatchNormalization()(x)
    x = Dense(32,   activation='relu', kernel_regularizer=REG)(x); x
       = Dropout(0.1)(x); x = BatchNormalization()(x)
    x = Dense(16,   activation='relu', kernel_regularizer=REG)(x); x
       = Dropout(0.1)(x); x = BatchNormalization()(x)
    out = Dense(N_CLASSES, activation='softmax', name='classes')(x)
    m = Model(inp, out, name='Classifier_EXACT')
    m.compile(optimizer='adam', loss='categorical_crossentropy',
       metrics=['accuracy'])
    return m

def build_detector_exact(trained_classifier, n_hidden_1=32,
   n_hidden_2=16, n_hidden_3=16):
    trained_classifier.trainable = False
    inp = Input(shape=(IN_FEAT_CSI,), name='input_bits')
    classes = trained_classifier(inp, training=False)

    cc1 = concatenate([classes, inp]); t1 = BatchNormalization()(cc1
       )
    t1  = Dense(n_hidden_1*4, activation='relu', kernel_regularizer=
       REG)(t1)

    cc2 = concatenate([classes, t1]);   t2 = BatchNormalization()(cc2
       )
    t2  = Dense(n_hidden_2*4, activation='relu', kernel_regularizer=
       REG)(t2)

    cc3 = concatenate([classes, t2]);   t3 = BatchNormalization()(cc3
       )
    t3  = Dense(n_hidden_3*4, activation='relu', kernel_regularizer=
       REG)(t3)

    cc4 = concatenate([classes, t3]);   t4 = BatchNormalization()(cc4
       )
    out_bits = Dense(payloadBits_per_OFDM, activation='sigmoid',
       name='bits')(t4)

    # *** Added metric bit_acc (no architecture change) ***
    m = Model(inp, out_bits, name='Detector_EXACT')
    m.compile(optimizer='adam', loss='mse', metrics=[bit_err,
       bit_acc])
    return m

# -------------------------
# Training knobs (yours)
# -------------------------
BATCH            = 100000
EPOCHS_CLS       = 30
EPOCHS_DET       = 30
STEPS_PER_EPOCH_CLS = 50
VAL_STEPS_CLS       = 25
STEPS_PER_EPOCH_DET = 50
VAL_STEPS_DET       = 25

TEST_SNRS    = [0, 5, 10, 15, 20]
TEST_SAMPLES = 100000

# -------------------------
# Per-channel validation loss callback (one figure)
# -------------------------
class PerChannelValLoss(Callback):
    def __init__(self, model, val_sets: dict, steps_per_val: int =
        5):
        super().__init__()
        self.model = model
        self.val_sets = val_sets          # dict: name -> (generator,
            steps)
        self.history = {k: [] for k in val_sets.keys()}

    def on_epoch_end(self, epoch, logs=None):
        # evaluate loss only (index 0)
        for name, (gen, steps) in self.val_sets.items():
            res = self.model.evaluate(gen, steps=steps, verbose=0)
            loss = float(res[0]) if isinstance(res, (list, tuple, np
                .ndarray)) else float(res)
            self.history[name].append(loss)

# -------------------------
# Train Classifier
# -------------------------
classifier = build_classifier_exact()
ckpt_cls = ModelCheckpoint(os.path.join(OUTDIR, 'cls_best.weights.h5
    '),
                           monitor='accuracy', mode='max',
                           save_best_only=True, save_weights_only=
                               True, verbose=1)
classifier.fit(
    training_class_gen(BATCH, 20),                    # mixed SNR
        (0..20)
```

```python
    steps_per_epoch = STEPS_PER_EPOCH_CLS ,
    epochs=      EPOCHS_CLS          ,
    validation_data=validation_class_gen(BATCH, 20),
    validation_steps=VAL_STEPS_CLS,
    callbacks=[ckpt_cls, ReduceLROnPlateau(monitor='val_loss', factor=0.5, patience=10, verbose=1)],
    verbose=2
)
classifier.load_weights(os.path.join(OUTDIR, 'cls_best.weights.h5'))

# -------------------------
# Train Detector (cascade) + record training accuracy & val loss
# -------------------------
detector = build_detector_exact(classifier, n_hidden_1=32, n_hidden_2=16, n_hidden_3=16)
ckpt_det = ModelCheckpoint(os.path.join(OUTDIR, 'det_best.weights.h5'),
                           monitor='val_bit_err', mode='min',
                           save_best_only=True, save_weights_only=True, verbose=1)

# Build per-channel validation generators for the callback (at train SNR ~ mixed: pick 20 dB or any)
VAL_SNR_FOR_CURVES = 20
val_sets = {
    "Rayleigh": (validation_gen_fixed(3000, VAL_SNR_FOR_CURVES, 'rayleigh'), 5),
    "Rician"  : (validation_gen_fixed(3000, VAL_SNR_FOR_CURVES, 'rician'),   5),
    "WienerII": (validation_gen_fixed(3000, VAL_SNR_FOR_CURVES, 'winnerii'), 5),
    "AWGN"    : (validation_gen_fixed(3000, VAL_SNR_FOR_CURVES, 'awgn'),     5),
}
pcvl_cb = PerChannelValLoss(detector, val_sets, steps_per_val=5)

hist_det = detector.fit(
    training_gen(BATCH, 20),                        # mixed SNR (0..20)
    steps_per_epoch = STEPS_PER_EPOCH_DET ,
    epochs=EPOCHS_DET ,
    validation_data=validation_gen(BATCH, 20),
    validation_steps = VAL_STEPS_DET ,
    callbacks=[
        ckpt_det,
        ReduceLROnPlateau(monitor='val_bit_err', factor=0.5,
           patience=20, mode='min', verbose=1),
        EarlyStopping(monitor='val_bit_err', patience=60,
           restore_best_weights=True, mode='min', verbose=1),
        pcvl_cb,    # << per-channel validation loss recorder
    ],
    verbose=2
)
detector.load_weights(os.path.join(OUTDIR, 'det_best.weights.h5'))

# --------------------------
# Save & plot: TRAINING ACCURACY (detector) and VALIDATION LOSS (
   mixed)
# --------------------------
epochs_axis = np.arange(1, len(hist_det.history['loss']) + 1, dtype=
   np.int32)

train_acc_curve = np.array(hist_det.history.get('bit_acc', []),
   dtype=np.float32)
val_loss_curve  = np.array(hist_det.history.get('val_loss', []),
   dtype=np.float32)%
```

Listing 6.1: Simple universal DNN architecture (PyTorch).