High-speed Machine Learning-enhanced Receiver for Millimeter-Wave Systems

Dolores Garcia[†⋆], Rafael Ruiz[†⋆], Jesus O. Lacruz[†], Joerg Widmer[†]
[†] IMDEA Networks Institute, Madrid, Spain
[⋆] Universidad Carlos III de Madrid, Madrid, Spain
E-mails: [†]{firstname.lastname}@imdea.org

***Abstract*—Machine Learning (ML) is a promising tool to design wireless physical layer (PHY) components. It is particularly interesting for millimeter-wave (mm-wave) frequencies and above, due to the more challenging hardware design and channel environment at these frequencies. Rather than building individual ML-components, in this paper, we design an entire ML-enhanced mm-wave receiver for frequency selective channels. Our ML-receiver *jointly* optimizes the channel estimation, equalization, phase correction and demapper using Convolutional Neural Networks. We also show that for mm-wave systems, the channel varies significantly even over short timescales, requiring frequent channel measurements, and this situation is exacerbated in mobile scenarios. To tackle this, we propose a new ML-channel estimation approach that *refreshes* the channel state information using the guard intervals (not intended for channel measurements) that are available for every block of symbols in communication packets. To the best of our knowledge, our ML-receiver is the first work to outperform conventional receivers in general scenarios, with simulation results showing up to 7 dB gains. We also provide an experimental validation of the ML-enhanced receiver with a 60 GHz FPGA-based testbed with phased antenna arrays, which shows a throughput increase by a factor of up to 6 over baseline schemes in mobile scenarios.**

***Index Terms*—Machine Learning, Millimeter Wave, Channel Estimation, Physical Layer, Equalization**

I. Introduction

Machine Learning (ML) technology has shown great advances over the last decade, mainly due to the abundant and affordable high-performance computation engines that allow fast training of large neural networks. In wireless systems, ML has been extensively used at high network protocol layers, for example for traffic prediction, blockage prediction or anomaly detection [1–3], where ML techniques often outperform classical algorithms. Recently, several works showcased that ML is also suitable to design components of the wireless physical layer (PHY) [4–12]. Their main motivation is that conventional PHY layers are composed of highly optimized but separate signal processing blocks such as modulator/demodulator, equalizer, demapper, etc. This block-wise design facilitates implementation and modularity, but does not always result in optimum performance. Furthermore, these blocks are often designed under the assumption of linear processes without considering potential non-linearities and residual errors due to the interaction among blocks. Also channel tracking over short timescales is usually not a main concern in these designs. Such issues become more relevant at higher frequencies, in particular the millimeter-wave (mm-wave) band and at THz frequencies, where the high frequency induces channel fluctuations even in static scenarios, device complexity increases, and miniaturization makes hardware imperfections more complex and harder to model.

Designing high-performance RF components is more complex at mm-wave frequencies, and ML-based systems can inherently learn how to cope with imperfections. Achieving high gains with small mm-wave components is difficult, and they are more susceptible to noise and deviations. Giga-sampling rate AD/DA converters for wide-band channels often have limited resolution to avoid excessive complexity and power consumption, which introduces non-negligible quantization effects [13]. The directional antenna arrays used in mm-wave systems to compensate for the high path loss require complex calibration methods to ensure matching between phase shifters and amplifiers of antenna elements [14, 15]. Future THz-frequencies systems further exacerbate this situation, making ML a highly attractive tool to design and optimize the PHY.

While several prior works demonstrate that using ML for the wireless PHY is beneficial [7, 8, 10, 11, 16], only few have studied full ML-based receiver architectures [12, 17, 18]. These works focus only on sub-6 GHz Orthogonal Frequency-Division Multiplexing (OFDM) systems. DeepRX [17] and its MIMO extension [18] are Convolutional Neural Network (CNN)-based receivers for OFDM that use the received signal, raw channel estimate, and received pilots as inputs. The receiver network perform channel estimation, equalization and demapping jointly. In [12], a similar design for MU-MIMO is presented, that improves second order statistics of the channel estimation to jointly optimize equalization and demapping. These architectures substitute one or multiple blocks of the PHY. While showing potential for performance gains over traditional receivers, they only outperform conventional minimum mean square error (MMSE) baselines under interference.

For the higher frequency mm-wave systems we focus on, Single Carrier (SC) PHY designs become more practical due to OFDM's high peak-to-average power ratio [19, 20], which requires a more power-hungry power amplifier to maintain the same communication range [21]. OFDM systems also have strong spectral leakage, strict synchronization procedures, and are more sensitive to Carrier Frequency Offset (CFO), all of which are more difficult to manage at high frequencies. Given

the sparse mm-wave multi-path environment, SC systems perform well and have a less complex implementation [20].

We propose a complete SC mm-wave ML-enhanced receiver architecture (ML4RX) that embraces both hardware imperfections and channel variability. We first present an ML-enhanced receiver architecture that performs classical MMSE equalization [22] and only uses a CNN network to optimize the symbol demapper. This network takes a full block of data symbols and the estimated channel as inputs and then outputs the log-likelihood-ratio (LLR) of the transmitted bits. We then enhance the receiver architecture with an ML-based channel estimation and tracking mechanism. We base our design on two observations: i) the channel measurements degrade rapidly over time not only for mobile but even for *static* scenarios, and ii) SC mm-wave systems prepend and append Guard Intervals (GIs) to blocks of data symbols [23, 24]. Those help to implement low complexity frequency-domain equalization as well as to compensate for residual CFO and phase impairments. We can thus take advantage of these periodic GI sequences to enhance the channel estimate, which is (commonly) obtained only once from the preamble of a received packet. By introducing the GI information into our ML pipeline, we obtain (coarse) periodic channel updates without any additional overhead. The enhanced channel estimation procedure allows to interpolate the channel over time and provides updated information about changes in the multi-path profile. We feed both the channel estimate from the preamble and the coarse measurements from the GIs together with the received data to the ML-enhanced receiver to perform joint channel estimation, equalization, and demapping.

To train and validate our proposed receiver architecture, we generate datasets using the Quadriga [25] software, that allows to generate realistic channel impulse responses with time evolution. We also evaluate our system using real measurements with an FPGA-based testbed with 60 GHz phased antenna arrays [26]. As a critical requirement for a practical system, we demonstrate that the ML training does not need to cover the whole environment, i.e., training with some representative receiver trajectories in the environment is sufficient. The performance of the different experiments is evaluated using the bit error rate (BER) and throughput as metrics. The evaluation shows up to 7 dB gain for coded and uncoded data transmissions which results in throughput gains of up to 12% in static scenarios and 620% in mobile scenarios.

This paper makes the following contributions:

- Our design is the first ML-receiver pipeline that generally *outperforms even MMSE receivers with perfect channel knowledge*. We show that ML4RX's performance is closer to that of the optimal Maximum Likelihood Sequence Estimator (MLSE) receiver that has exponential complexity, while maintaining the low complexity of MMSE. The performance improvement is due to jointly optimizing the enhanced channel estimation together with equalization and soft demapping to obtain soft bit information that maximizes the BER. The improved channel estimate is obtained using a neural network that merges the high-resolution channel estimate from the preamble with the coarse periodic channel information from GIs.
- We evaluate the receiver using real-world measurements from an FPGA-based mm-wave transceiver platform. To our knowledge, our static and mobile experiments are the first real-world evaluation of an ML-enhanced mm-wave receiver, and they demonstrate that such a receiver architecture can significantly outperform conventional designs in practice. Overall, we believe that our design and experiments are crucial steps forward to bring ML-enhanced PHY architectures to real-world wireless systems.

The paper is organized as follows. In Section II we introduce the system model and baseline receivers. In Section III we present the ML4RX architecture and its complexity. Practical implementation aspects are discussed in Section IV. We compare the performance of our design to several baseline schemes by simulation and testbed experiments in Sections V and VI, respectively. We discuss the related work in Section VII. Finally, we provide concluding remarks in Section VIII.

II. System Model and Baseline Receiver

We first present the system model and standard baseline receiver designs, their channel estimation methods, and complexity.

A. System model

We consider an SC system and a frequency selective channel with inter-symbol interference (ISI). The received signal y_r is

$$y_r[n] = \sum_{l=0}^{M-1} h[l]x[n-l] + v[n] \quad n = 0, \dots N_D - 1 \quad (1)$$

where $y_r \in \mathbb{C}^{N_D}$ is the received data block of length N_D, $h[l] \in \mathbb{C}$ are the channel coefficients, M is the length of the channel impulse response, $x \in \mathbb{C}^{N_D}$ are the transmitted symbols, and v is thermal noise at the receiver.

B. Baseline receivers

At the receiver, communication packets are usually detected using a preamble composed of known sequences. The preamble is also used to estimate and correct CFO and to perform synchronization. Finally, specific parts of the preamble allow to estimate the channel and specifically to extract the Channel Impulse Response (CIR). Without loss of generality, in this paper we focus on mm-wave systems that use Golay sequences for this purpose, but the CIR can also be estimated with other sequences with suitable correlation properties.

1) Equalization: The estimated CIR $\hat{\mathbf{h}}$ is used for equalization. We consider both MMSE and MLSE.

• *Frequency domain MMSE equalization*: For a frequency selective channel, the equalization is usually performed in the frequency domain, as the implementation is less costly in hardware. In the frequency domain, Eq. (1) is equivalent to the product of the N_D-point discrete Fourier transform (DFT) of the transmitted signal and the channel response

$$y[k] = h[k]x[k] \quad k = 0, \dots, N_D - 1\ . \quad (2)$$

The MMSE equalization utilizes an estimate of the noise variance σ_n to calculate the Channel State Information (CSI)

$$CSI[k] = h[k]h^*[k] + \sigma_n \tag{3}$$

and therefore, the frequency domain equalization is obtained using the channel estimates $\hat{h}$ by computing

$$\hat{x} = \text{IDFT}_{N_D} \left\{ \frac{y[k]\hat{h}^*[k]}{CSI[k]} \right\} . \tag{4}$$

• *Maximum Likelihood Sequence Estimator*: While MMSE is used in practice due to its lower complexity, the optimum decoding algorithm is MLSE [27, 28], which solves the optimization problem

$$\min \sum_{n=1}^{N_D} \left| y_r[n] - \sum_{l=1}^{M} h[l]x[n-l+1] \right|^2 \tag{5}$$

to find the most likely transmitted sequence. This problem can be solved using the Viterbi algorithm [29, 30]. However, the complexity of the Viterbi algorithm is exponential with the length of the channel and is proportional to the constellation size. Even sphere decoding together with the Viterbi algorithm [31] has exponential worst case complexity.

2) Constellation demapping: The soft-decision approximate LLR method [32, 33] is used to extract soft bit information from the equalized symbols using the maximum a posteriori probability (MAP). The approximate LLR is obtained using the max-log-MAP, a well known approximation of the exact log-MAP demapping algorithm.

3) Guard intervals in SC high frequency communications: GIs are used to compensate phase impairments and residual CFO present in high frequency systems. GIs are interleaved with blocks of data symbols, acting as a prefix to simplify frequency domain equalization. In current high-frequency systems, the length of the data block (data symbols + GI) is limited to a few hundreds of symbols in order to allow for an efficient implementation of the Fast Fourier Transform (FFT). Ideally, the length of the GI (N_L) should be longer than the maximum delay spread of the channel to avoid ISI. Phase impairments are corrected by computing the phase difference between two consecutive GIs and then *correcting* the data symbols in between. Considering the relative short delay spread of mm-wave channels and that the GI is mainly used for phase estimation, a relatively short GI size is used in practice, to avoid excessive overhead.

C. Baseline receiver complexity

The complexity of MMSE frequency domain equalization scales with the data block size N_D as $\mathcal{O}(N_D^2)$. In contrast, MLSE scales with the data block size and the constellation order as $\mathcal{O}(N_D 2^{CM})$, where 2^C is the constellation size and M is the length of the channel impulse response. The demapping complexity scales as $\mathcal{O}(N_K 2^C)$, where $N_K = N_D - N_L$ is the length of the data symbols after GI removal. Therefore the total receiver complexity scales as $\mathcal{O}(N_D^2 + N_K 2^C)$ for the MMSE and $\mathcal{O}(N_D 2^{CM} + N_K 2^C)$ for the MLSE baselines.

III. ML4RX Receiver

We now detail the architecture of our ML-enhanced receiver that jointly optimizes the components of a receiver: channel estimation, equalization, and soft demapping. The network is trained to output LLRs that minimize the BER of the system and can be used in conjunction with a channel coding scheme.

We first consider a system with classical equalization, replacing the demapper with a CNN block. Then, we extend the system to use the GIs to obtain coarse-grained but up-to-date channel estimates over the course of a packet. The key intuition behind this is that the channel estimated from the GIs can be merged with the outdated channel estimate from the preamble to better track the evolution of the channel. Since this improved channel estimation happens jointly with equalization and demapping, the ML4RX receiver can learn the best joint channel estimation, equalization, and demapper for a given hardware and environment.

A. Training the ML4RX pipeline

The ML4RX receiver architecture is shown in Fig. 1. The components with trainable weights are shown in blue. The packets are sliced using a block separator, which splits the packet into data blocks. Then, the receiver processes each data block separately and jointly optimizes the channel estimation, equalization, phase correction, and demapping. As shown in the figure, the ML4RX receiver pipeline has three input arrays as follows:

- ts_r: are the received training symbols used to compute the CIR estimate in the time domain, $\hat{\mathbf{h}}$, with $\hat{\mathbf{h}} \in \mathbb{C}^{N_D \times 1}$.
- g_r: is the received GI for each data block. This is used to compute the *GI channel estimate* $\hat{\mathbf{h}}_l \in \mathbb{C}^{N_D \times 1}$, denoted with an l since its knowledge of the channel is limited by the size of the GI. The channel estimation using the GI is explained in Section IV-C.
- y_r: is the received data block, comprising the data symbols and again the GI g_r.

The outputs of the model are the estimated LLRs of each received symbol, $\mathbf{L\hat{L}R} \in \mathbb{R}^{N_K \times C}$, where C is the modulation order. The LLRs can be converted to bit probabilities by applying the sigmoid function $\hat{\mathbf{b}} = \text{sigmoid}(\mathbf{L\hat{L}R})$. The trainable parameters of the network θ are optimized to minimize the total binary cross-entropy (BCE):

$$\begin{aligned} \mathcal{L} = \sum_{l=0}^{N_K-1} \sum_{c=1}^{C} \mathbb{E}_{\mathbf{y_r}} [b_{l,c} \log(P_\theta(b_{l,c} = 1|\mathbf{y_r})) + \\ (1 - b_{l,c}) \log(1 - P_\theta(b_{l,c} = 1|\mathbf{y_r}))] \end{aligned} \tag{6}$$

where $P_\theta(\cdot|\mathbf{y}_r)$ is the posterior distribution of the bits given the received signal $\mathbf{y}_r$. The loss function is approximated using Monte Carlo sampling by taking batches of data

$$\begin{aligned} \mathcal{L} \approx -\frac{1}{B} \sum_{b=1}^{B} \sum_{l=0}^{N_K-1} \sum_{c=1}^{C} \mathbb{E}_{\mathbf{y_r}} [b_{l,c} \log(\hat{b}_{l,c}) \\ + (1 - b_{l,c}) \log(1 - \hat{b}_{l,c})] \end{aligned} \tag{7}$$

where B denotes the batch size.

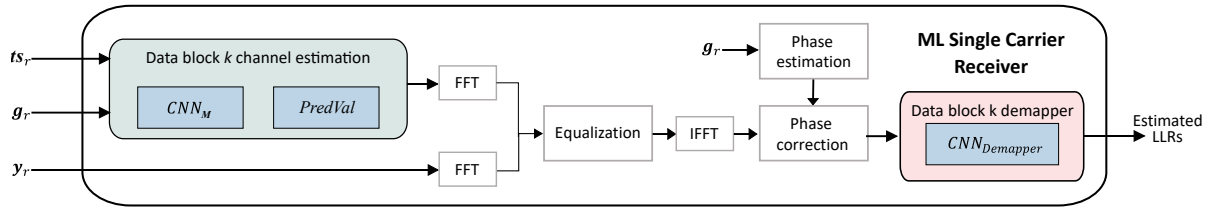


Fig. 1: ML receiver architecture

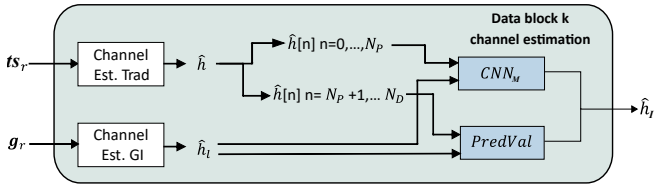


Fig. 2: ML-enhanced channel estimation architecture

B. ML-enhanced channel estimation

As presented in Section II, the channel estimate accuracy for data blocks near the preamble and then degrades over time. This is particularly pronounced in case of mobility or dynamic environments, but even static scenarios show perceptible fluctuations in multi-path gains over time. We now discuss the improved channel estimation block of the ML4RX receiver pipeline that merges the channel estimates from the preamble and the GIs as shown in Fig. 2. First, the channel estimates are obtained from the preamble and the GI. Since the GI is much shorter than the preamble, its channel estimate $\hat{\mathbf{h}}_l$ contains fewer channel coefficients than $\hat{\mathbf{h}}$. Also, for a GI of length N_L we only obtain $N_P = 0.5N_L$ channel coefficients, since the first coefficients are contaminated with symbols from the previous block. Thus, the preamble channel estimate is outdated but more accurate, whereas the GI channel estimate is coarse but more up-to-date.

The improved channel estimation sub-block has two parts: i) merging the coefficients seen by both the preamble and the GI channel estimates up to N_P, and ii) predicting the coefficients that are not seen by the GI from the other data. To merge the coefficients seen by both estimates, we design a CNN denoted by CNN_M. The CNN takes as inputs $\hat{\mathbf{h}}_l \in \mathbb{C}^{N_P \times 1}$ and $\hat{h}[n] \in \mathbb{C}$ with $n = 0, \ldots, N_P$ and outputs $\hat{\mathbf{h}}_I[n] \in \mathbb{C}$ with $n = 0, \ldots, N_P$. Then, a prediction block formed by linear layers $PredVal$ is designed to get up-to-date estimates of the channel coefficients not seen by the GIs. The CNN is a single 1D convolutional layer to increase explainability, as then the weights of the layer can directly be linked to the coefficient that the network is assigning to both the channel estimate from the GI and the preamble. The $PredVal$ has as input $\hat{\mathbf{h}}_l \in \mathbb{C}^{N_P \times 1}$ and $\hat{h}[n] \in \mathbb{C}$ with $n = N_P + 1, \ldots, N_D$ and outputs $\hat{\mathbf{h}}_I[n] \in \mathbb{C}$ with $n = N_P{+}1, \ldots, N_D$, with output dimension $(N_D{-}N_P){\times}1$. Finally, the channel estimate vectors from both sets of layers are concatenated to form $\hat{\mathbf{h}}_I$.

For equalization and phase correction, the results of the improved channel estimation sub-block, $\hat{\mathbf{h}}_I$, and the received symbols, $\mathbf{y}_r$ are converted to the frequency domain using FFT and multiplied as in Eq. (2) which are ultimately converted back to time-domain by computing an inverse Fourier transform. A phase estimate is obtained from the received GI and the known sequence of the GI as $\phi_{\text{error}} = \angle(\mathbf{g_r} \cdot \text{GI})$, where $\angle$ is the angle of the complex number in polar coordinates. The phase error between two consecutive GIs is $\delta_\phi = \phi(1) - \phi(0)$, where $\phi(i)$ is the phase error of a given GI. Therefore, the phase to correct each symbol is estimated by interpolating between phase errors of different GIs, where we assume the phase error is measured approximately at the center of the GIs.

$$\mathbf{\Phi} = \phi(0) - \delta_\phi(0)(1 - \frac{1}{N_L}(\frac{N_L}{2} + \mathbf{k})) \tag{8}$$

where $\mathbf{k} = [0, \ldots, N_D]$. Then, the equalized symbols $\mathbf{s}_e$ are phase corrected as $\mathbf{s}_{pe} = \mathbf{s}_e e^{-i\mathbf{\Phi}}$.

C. ML-enhanced demapper

Following the equalization and phase correction, the symbols are inputted to the demapper block. As a result of imperfect channel estimation, residual CFO and phase noise, the symbols are distorted further than just thermal noise. A traditional demapper, as described in Section II, is applied in a per symbol manner, as the equalization should take care of the ISI. As a consequence, these residual effects are not considered and the demapper observes them as noise and distortions in the data. In contrast, our architecture jointly processes the data block of equalized symbols and this allows to take into account and correct non-linearities, residual ISI and other residual effects described above. The inputs to the $\text{CNN}_{\text{Demapper}}$ are the real and imaginary parts of the vector of equalized symbols s_{pe} of dimension $2 \times N_K$, after GI removal. The outputs are the predicted LLRs of dimension $N_K \times C$. The demapper sub-block is shown in Fig. 3. The architecture is formed by ResNets [34] with 1D convolutions and kernels of fixed size 3. We performed experiments with variable kernel sizes but did not observe any significant gains.

D. Complexity

The complexity of the ML4RX channel estimation block is of order $\mathcal{O}(N_D^2)$, where CNN_M has lower complexity $\mathcal{O}(N_D)$ and the $Predval$ block has complexity of order $\mathcal{O}(N_D^2)$. Therefore, the equalization of the ML4RX receiver has the same order of complexity as the MMSE equalization that scales with the data block size, N_K as $\mathcal{O}(N_K^2)$. The $\text{CNN}_{\text{Demapper}}$ complexity scales as $\mathcal{O}(N_K 2^C)$, which is equivalent to the order of complexity of the log-MAP demapper. Therefore the total ML enhanced receiver complexity scales as $\mathcal{O}(N_D^2 + N_K 2^C)$, which is equal to the order of complexity of the MMSE baseline with the log-MAP demapper.

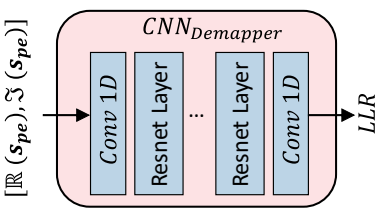


Fig. 3: CNN Demapper subblock

TABLE I: CNN Demapper

	$\mathbf{CNN_{Demapper}}$	
Input size	$B \times 2 \times N_K$	
Parameters	Filters	Kernel
Conv 1D	32	3
Resnet 1	64	3
Resnet 2	64	3
Resnet 3	32	3
Conv 1D	C	1
Output size	$B \times C \times N_K$	

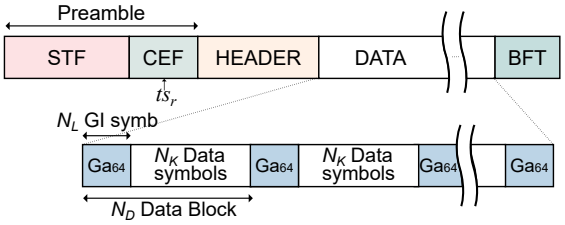


Fig. 4: IEEE 802.11ad frame structure

IV. Implementation

In this section, we cover the implementation details and practical aspects of the receiver design.

A. Training strategy

The training is performed in two parts. First, the parameters from the $\text{CNN}_{\text{Demapper}}$ sub-block are trained while the parameters from the improved channel estimation sub-block are fixed, so that a good demapper is obtained for a given equalization. Then the required gradients are inverted. The reason for having two training steps is that the demapper depends on the resulting equalized symbols and if these are updated at every training step, the demapper sub-block does not converge to the best solution.

Without loss of generality, let us consider the IEEE 802.11ad/ay standards [32, 33] as an example to calculate the size of the training model. The frame structure for these standards is shown in Fig. 4, where the block size N_D is 512, comprising $N_K = 448$ data symbols and $N_L = 64$ symbols for the GI. Using those parameters, the model size of the architectures presented in Section III is 200 kB. Such a small model fits even into low-end devices.

B. CIR estimation in the preamble

Current systems using standards such as 802.11ad/ay use complementary Golay sequences in the Channel Estimation Field (CEF). The training symbols ts_r in the CEF are used to estimate the CIR. To this end, CIR estimation takes advantage of the auto-correlation properties of complementary Golay sequences [35]. The sum of the auto-correlation of a pair of complementary Golay sequences Ga_N and Gb_N of length N is the delta function $\delta[n]$. This property makes these sequences very suitable for multi-path estimation in noisy environments. The CIR for each pair of complementary Golay sequences d is given by:

$$h_{G,d}[i] = \frac{1}{N}\sum_{n=0}^{N-1} r_{CEF}[i+n] \times \mathrm{G}_N^*[n] \quad \forall\ d = 1, ..., D \quad (9)$$

We denote by r_{CEF} the CEF field of the received frame. D is the number of repetitions of Golay sequences in the CEF and $\{\cdot\}^*$ is the complex conjugate operator. We use Eq. (9) to determine $h_{Ga,d}$ and $h_{Gb,d}$ for Ga and Gb, respectively. Finally, the estimated CIR is obtained by adding and averaging $h_{\text{Ga},d}$ and $h_{\text{Gb},d}$ as follows:

$$\hat{\mathbf{h}} = \frac{1}{D}\sum_{d=1}^{D}(\mathbf{h}_{\text{Ga},d} + \mathbf{h}_{\text{Gb},d}) \ . \quad (10)$$

C. CIR estimation using guard intervals

Apart from the functionality of the GI described in Section II-B, the received GI ($\mathbf{g_r}$) can be used to obtain a CIR estimate. Since there is a GI in every data block, this permits obtaining CIR estimates at regular short intervals. The number of channel coefficients to be estimated, N_P, is limited by half of the length of the GI (N_L), which in turn limits the accuracy of the estimated CIR since only N_P taps can be observed. N_P may be smaller than the actual number of channel coefficients, if the delay spread of the channel is longer than half a GI. Also, note that in IEEE 802.11ad/ay systems, the GI corresponds to Ga_{64} Golay sequences (without complementary sequences), which makes channel estimation noisier and more challenging.

The channel estimate is obtained as follows. The n-th element of received GI can be written as

$$g_r[n] = \sum_{l=0}^{N_P} h[l]g[n-l] + v[n] \quad n = 0, \dots N_L \quad (11)$$

where $g[n]$ are GI elements, $\mathbf{h_l}$ is the vector of limited channel coefficients to be estimated and v is the thermal noise from the receiver and inter-symbol interference in case N_P is smaller than the real number of paths. The first received element we can use is $g_r[N_P]$ as the first N_P guard symbols could include contributions from the previous data blocks due to multipath components. In matrix form, the system of equations can be written as

$$\mathbf{g}_r = \mathbf{G}\mathbf{h_l} + w \quad (12)$$

where $\mathbf{G}$ is the matrix with the m-th row equal to $\mathbf{G}_m = [g[N_P + m - 1], \dots, g[m-1], g[m]]$. In order to obtain the estimated channel $\hat{\mathbf{h}}_l$ we solve the least-squares problem

$$\hat{\mathbf{h}}_l = \min_{h_l} ||\mathbf{g}_r - \mathbf{G}\mathbf{h_l}|| \ . \quad (13)$$

V. Evaluation by Simulation

We now evaluate the proposed receiver architecture against the baselines presented in Section II in a simulation setup. This allows us to thoroughly study its performance under different conditions and separate the impact of the different components while having ground truth for the channel realizations. First, we introduce the simulation setup and explain the dataset generation process. Then, we evaluate the results.

A. Simulation setup

In order to evaluate the performance of our receiver, we generate channel realizations using the Quadriga software [36]. Quadriga allows to generate realistic channel impulse responses with time evolution. For the simulations, we focus on the downlink *mmMagic* Urban Microcell Non-Line of Sight scenario [25]. The transmitter and receiver have dipole antennas and the center frequency is 60 GHz.

To be able to compare the ML-enhanced receiver to the maximum likelihood scheme, MLSE, the coefficients of the channel are generated with only 4 taps. A tap is an element of the channel's impulse response. If the number of taps is larger, the trellis has more than 2^{20} states and the complexity becomes too high to determine the MLSE solution even on a large GPU server. A trellis is a directed graph that describes systems with memory and it is used in the Viterbi algorithm to solve the maximum likelihood MLSE minimization problem. The positioning of these taps is defined by their delay spread and angular spread. The angular spread has values around 20-90° and the delay spread has a log-normal distribution with a mean of 16 ns and standard deviation of 2 ns, which are typical values for mm-wave. We consider the following scenarios:

1) NLOS comparison to baseline schemes: The receiver is placed at a distance of 2 m from the transmitter and follows a linear trajectory away from the transmitter with an angle of $\pi/8$. The channel model is the Quadriga model for NLOS. The channel at the initial point of the trajectory, $\mathbf{h}_0$, is considered as the channel estimated by the preamble. A second channel, $\mathbf{h}_{0.5ms}$, obtained using the continuous-time evolution in Quadriga, is obtained 0.5 ms later ($\sim$ half the coherence time). The following data blocks are processed with $\mathbf{h}_{0.5ms}$ and a channel estimate is obtained using the GI $\hat{\mathbf{h}}_{0.5ms,l}$. Separate training sets are produced for each Signal-to-Noise Ratio (SNR). We generate 1000000 pairs of channel realizations and each dataset consists of randomly generated bits modulated with 16 QAM and propagated through these channels. We study the performance of uncoded and coded data at different rates.

2) NLOS evolution over time: To study the evolution of the BER using the different methods over time, we fix the SNR and generate sets of channel realizations for up to 1 ms, as after 1 ms the BER of the data blocks equalized with the channel estimated from the preamble of 1 ms before decreases by several orders of magnitude. The receiver follows random linear trajectories away from the transmitter starting at a distance from 3 to 10 m. Again, we generate 1000000 sets of channel realizations, and send randomly generated bits modulated with 16 QAM through these channels.

3) Large number of paths: We repeat the experiment with sets of channel realizations with 10 taps, where the MLSE baseline can no longer be evaluated.

B. Training

The training is carried out using the Adam optimizer [37], starting from a random initialization of the $\text{CNN}_{\text{Demapper}}$ sub-block. For the CNN_M, we consider an initialization that starts with the preamble channel only. The $PredVal$ is initialized with zeros. The starting learning rate is 10^{-2} and we set an exponential decay that acts every epoch with a multiplicative factor of 0.95. For all experiments, 20% of the data is used for validation. The training is carried out with batch size $B = 512$ and for 20 epochs as we did not observe improvements for longer runs. For the training, we use an A100 GPU server with 40 GB of vRAM.

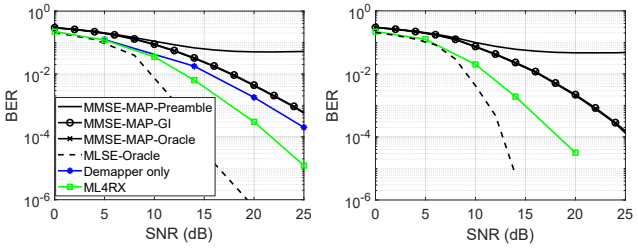
C. Simulation results

1) Comparison to the MMSE and MLSE baselines: For the first scenario, we benchmark our ML-enhanced receiver against the two baselines presented in Section II with different channel estimation variants. The first baseline is the MMSE equalization with the outdated channel from the preamble, $\mathbf{h}_{0ms}$ and log-MAP demapping (MMSE-MAP-Preamble). The second baseline is the MMSE equalization with perfect channel knowledge for each time instant (we refer to this as the oracle channel) plus the log-MAP demapper (MMSE-MAP-Oracle). The third one is the MMSE equalization with the channel estimated from the GI, and log-MAP demapping (MMSE-MAP-GI). The fourth is the MLSE equalization with perfect channel knowledge together with the approximate log-MAP demapper (MLSE-Oracle). Fig. 5a shows the BER of these receiver methods for different SNRs.

From the results we derive two conclusions. First, not updating the channel information results in a significant performance degradation over time, as can be seen by comparing MMSE-MAP-Preamble to the MMSE-MAP-Oracle. In this scenario, MMSE-MAP-GI shows the same performance as MMSE-MAP-Oracle since the delay spread of the channel is largely within the GI, allowing to estimate all important components in an overdetermined system. The second observation is that the ML-enhanced receiver ML4RX outperforms even the MMSE-MAP-Oracle by up to $\sim$ 7 dB, and is part-way to the optimal MLSE-Oracle baseline. The MLSE-Oracle has optimum performance, outperforming the MMSE baselines as well as ML4RX. In order to isolate the gains of the improved channel estimation block, we also compare the ML4RX curve to that of the Demapper only architecture with MMSE equalization. We observe up to ~ 5 dB improvement when using the full ML4RX architecture.

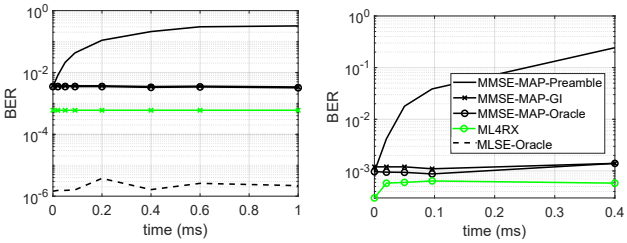
2) Soft information: Fig. 5b shows the results using the same baselines with LDPC coded data with rate $3/4$. The ML4RX receiver shows significant gains compared to the traditional receiver, with up to 6 dB gains for both rates, and is closer to the optimal performance of MLSE. The results with other code rates, not included in the figure, are similar. For example, with code rate $1/2$ the ML4RX receiver has up to 7 dB gain compared to the MMSE baselines and performs close to MLSE. These results shows that the soft bit information generated by the receiver pipeline is compatible with standard channel coding [38].

3) Evolution over time: As can be observed in Fig. 6a, the BER obtained using the channel estimation from the preamble at time zero degrades quickly over the 1 ms scale. In order to



(a) Uncoded (b) LDPC with code rate $3/4$

Fig. 5: BER achieved by ML4RX and the baselines



(a) 4-tap NLOS channel (b) 10-tap NLOS channel

Fig. 6: Evolution of the BER over time (SNR = 20dB)

keep the initial BER obtained at time zero, the channel would need to be sampled very frequently ($<$ 0.02 ms). Sampling every 0.02 ms would incur a large communication overhead of at least $\sim 10\%$. Under 1 ms, the channel changes due to variations in the path gains but there are no changes observed in the Angle of Arrival (AoA). Using the ML4RX receiver that utilizes the GI estimations, we can improve the performance over the MMSE-MAP-Oracle without incurring any overhead.

4) Large number of paths: Fig. 6b shows the results for the corresponding experiment with a longer channel response. After measuring the channel at time 0 ms, the BER obtained using the MMSE-MAP-Preamble quickly degrades by several orders of magnitude. The GI channel estimation is worse as we are trying to estimate more coefficients and the system in Eq. (12) becomes less overdetermined in this case. ML4RX again outperforms all MMSE baselines, including the MMSE-MAP-Oracle baseline with perfect channel information, by at least a factor of 2 to 4 in terms of BER.

VI. Experimental Evaluation

Although simulations offer important insights into the behavior of a system under controlled conditions, they cannot cover all the effects of real systems. For example, the simulated channel realizations do not have spatial coherence since they are generated from generic distributions in Quadriga. Having spatial coherence is important, as it allows the ML-enhanced demapper to learn about a certain environment. Additionally, in simulations, the static scenario does not present *any* channel variations over time. However, on real hardware, channel measurements vary over time, even for static scenarios. This is mainly due to quantization effects, imperfect symbol sampling, temperature variations, just to name a few. Furthermore, under mobile scenarios, it is nearly impossible to repeat the exact same trajectory, where even vibrations play a major role considering the wavelength of mm-wave frequencies.

A. Experimental platform

As basis for our experimental evaluation, we use the high-performance mm-wave FPGA-based platform from [26], whose implementation is freely available as open-source. The platform features an FPGA-based baseband processor capable of handling 2 GHz of bandwidth per channel and 60 GHz phased antenna arrays [39], as shown in Fig. 7. We use the testbed in a memory-based configuration to transmit stored

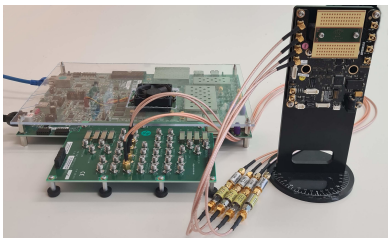


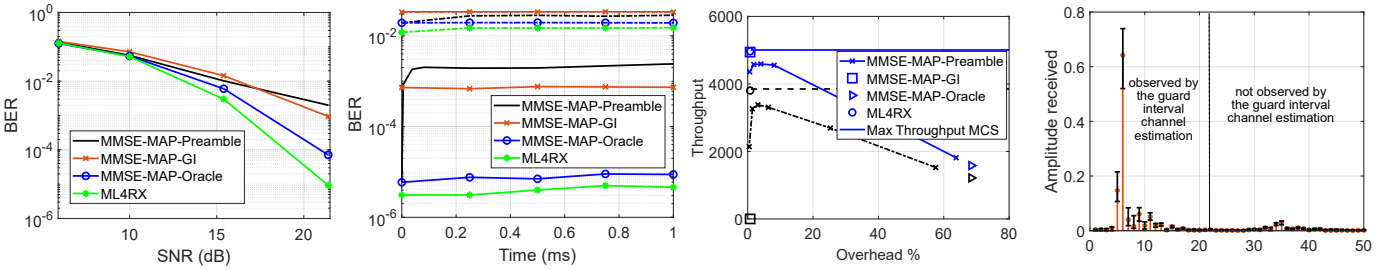
Fig. 7: Wideband mm-wave experimentation platform

packets over-the-air, and at the receiver store the received samples in the on-board memory to analyze them in post-processing.

B. Experimental results

a) Static deployment: First, we consider a scenario where the transmitter and receiver are static at a distance of 4 m in an indoor room. Over time, the channel varies due to changes in path gains [40–42] and internal noise in the devices. We first evaluate the uncoded achievable BER over all SNRs up to 0.1 ms after the preamble. For oracle channel knowledge, we insert one full preamble per data block for accurate channel estimation. This corresponds to very small and thus inefficient packets, which would result in 70% preamble overhead, 15% header overhead, and only 15% available for data. The results are presented in Fig. 8a. The BER obtained with the channel estimates from the GI, MMSE-MAP-GI, has worse performance than the other baselines. The ML4RX receiver outperforms the MMSE-MAP-Oracle baseline by $\sim$ 4 dB and the MMSE-MAP-Preamble by up to $\sim$ 8 dB, and the largest gains are obtained at higher SNRs. We believe that this is due to the importance of having up-to-date knowledge of the relevant paths that have the largest variability as seen in Fig. 8d, while being able to access the information from the outdated channel estimation which includes all paths and therefore is still relevant. The naive baseline of considering the first channel coefficients from the GI and the non-observed channel coefficients from the outdated preamble gave worse BER than both the MMSE-MAP-GI and the MMSE-MAP-Preamble and, for this reason, has not been included.

We consider an SNR of 20 dB and study the variation of the BER over time for the different receiver baselines up to 1 ms. We observe that the MMSE-MAP-Preamble degrades quickly. The ML4RX receiver improves the performance over the oracle and this is maintained over the whole 1 ms. We



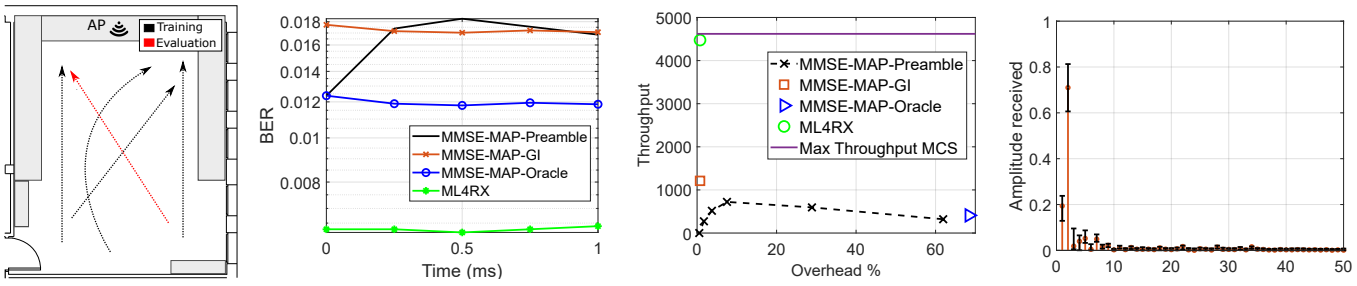
(a) BER of fixed transmitter and receiver over variable channel

(b) BER for 20 dB (solid) and 12 dB (dashed)

(c) Throughput for 20 dB/MCS 12.1 (blue) and 12 dB/MCS 11 (black)

(d) Error bars presenting the absolute maximum change over 0.5 ms.

Fig. 8: Results of the static experiments on FPGA test bed



(a) Trajectories

(b) BER variation over 1 ms for 12 dB

(c) Throughput for 12dB/MCS 12 (16QAM LDPC rate 3/4)

(d) Error bars presenting the absolute maximum change over 0.5 ms.

Fig. 9: Results of the mobility experiments on FPGA test bed

present the achievable throughput using the different baselines for an LDPC code rate of $13/16$ in Fig. 8c. The low overhead and low BER performance of the ML4RX receiver results in throughput improvements over all baseline schemes. In this high-SNR scenario, the MMSE-MAP-GI is quite accurate with low probability of packet error and same low overhead as ML4RX, and their throughput is thus very similar.

However, for a more complex scenario at 12 dB, the performance of MMSE-MAP-GI degrades drastically. Figs. 8b and 8c present the evolution over time of the uncoded BER and the throughput achieved for the different baselines using MCS 11, plotted in black. In this scenario, the BER of MMSE-MAP-GI is significantly higher than that of any other scheme. In contrast, the ML4RX receiver still outperforms the MMSE baselines including the oracle. In addition, the ML4RX matches the maximum throughput achievable at this MCS, outperforming by 13% the baseline with the best overhead configuration for this scenario. Finally, Fig. 8d shows the channel variations despite the scenario being static.

b) Mobile deployment: To evaluate the performance of the ML receiver under mobility, we deploy two nodes in a 7x7m room. While the receiver is static, the transmitter moves through the room while it sends data for 12 seconds for each measurement. We perform 5 different trajectories and we use 4 for training and 1 for evaluation, as shown in Fig. 9a.

The results for the mobile deployment shown in Fig. 9 follow the trend of the static scenario, but with an overall higher BER. However, for this dynamic case, the ML4RX significantly outperforms the MMSE baselines. This is reflected in a throughput improvement of 620% over the baselines as can be seen in Fig. 9c. This is due to the 50% improvement in BER which results in a lower packet error probability at this rate. Due to the slow speed, the channel variations over time observed in Fig. 9d are of the same order as those seen for the static scenario in Fig. 8d for the same timescale. We again remark that the training does not need to cover the whole environment for the receiver to achieve this performance and training and evaluation trajectories differ. This is critical to ensure the practicality of our design.

VII. Related Work

A. ML for the PHY

Different works have proposed to use ML to replace one or multiple blocks of the PHY, showing that it can not only reduce the complexity of certain methods but also improve the performance. For example, a neural network for soft demodulation can achieve good performance at reduced computational complexity compared to the traditional log-MAP demapping [7]. In [8], the authors propose a CNN network for equalization and obtain a lower error vector magnitude than the recursive least squares and multi-modulus algorithms. In [16], the channel estimation and symbol detection blocks for an OFDM system are substituted by ML designs, which helps with limited pilots and CP removal.

Closest to ML4RX are the works that implement ML-based receivers. [43] introduces a Deep-Learning based OFDM receiver. They replace the classical main units of an OFDM receiver with neural network counterparts. In addition, the

authors evaluate the system with real and simulated data and provide an FPGA implementation of the receiver. Similarly, DeepRX [17] presents a fully convolutional neural network architecture for a SIMO system. The inputs to their system are the raw channel estimates as well as the pilot positions and the received symbols of a transmission time interval for all subcarriers. These are used to estimate LLRs at the receiver. Results show that their architecture performs as well as the MMSE equalization with perfect channel information and it outperforms MMSE with imperfect channel information. Additionally, they outperform the baseline in scenarios with inter-cell interference. Both, our approach and DeepRX, have the same complexity as MMSE for SC and OFDM respectively, but the larger DeepRX architecture results in a larger overall model.

In [18], DeepRX is extended to support multi-input multi-output (MIMO). They introduce new layers to preprocess the received symbols and the channel estimates and explore two transformations to preprocess the data, a maximum ratio combining multiplicative approach and a fully learnt multiplicative transformation. In almost all channel models they show similar performance to the MMSE baseline with perfect channel knowledge. Finally, [12] presents an ML-enhanced receiver architecture for multi-user MIMO OFDM systems. They jointly optimize a neural network block to improve the channel error statistics and a demapper neural network, while performing classical channel estimation and equalization. Their architecture has better performance than the MMSE baseline with imperfect channel knowledge and for very high speeds is better than the MMSE with perfect channel information.

However, none of the above mentioned works tackle the problem of frequency selective channels for SC systems, and specifically, apply ML for the PHY for wide-band high-frequency communications. In addition, most of the them are evaluated via simulations rather than experimental evaluations and none of them address mobility cases, and in many cases only have the same performance as MMSE with perfect channel information, whereas MM4RX outperforms it.

B. Improved channel prediction

In OFDM, the cyclic prefix is usually an unknown sequence and the channel is estimated using pilots inserted in the frequency domain. However, the work in [44] proposes to modify the standard and insert a pseudo-random sequence in place of the cyclic prefix. They do this to improve the robustness of the channel estimation in rapidly fading channels while accepting the resulting ISI due to the lack of a cyclic prefix. In contrast to our work, they consider delay spreads corresponding to 4 to 8 taps which can easily be estimated inside the GI.

In [45], the authors consider the time-frequency response of fast fading communication channels as a two-dimensional image for an OFDM system. They show that they can interpolate the unknown values of the channel response using the information obtained from pilots better than MMSE with imperfect channel knowledge. More recently, in [46] the authors attempt to interpret Deep Learning (DL) based channel estimation architectures under linear, nonlinear and inaccurate channel statistics. They show, using the mean squared error to the perfect channel, that the DL estimator is close to the MMSE performance under linear systems and has better performance under non-linear systems. However, they do not present results showing how this affects the BER performance. In [47], the authors propose a DL based channel estimation algorithm for MIMO OFDM systems. Their network is based on 2D CNNs and LSTM layers. They train their architecture by optimizing the mean squared error (MSE). They show that with their improved channel estimation the performance improves over the MMSE with imperfect channel knowledge for speeds of 50 and 300 km/h and that their normalized MSE is lower than different baseline algorithms like MMSE.

Overall, these works have shown that improving the MSE is possible using DL techniques to improve channel estimation in OFDM systems. However, these works show less gains than the ML-based receivers presented in [17], where the receivers achieved the same performance as MMSE with perfect channel knowledge. However, since these works are not tested on the same dataset, these results are hard to compare. Our work differs from the above in that we jointly optimize the channel estimation with equalization and demapping.

VIII. Conclusion

We designed and implemented an ML-enhanced mm-wave receiver pipeline that jointly optimizes the ML sub-blocks for channel estimation, equalization, and demapping. The channel estimation takes into account additional periodic channel estimates obtained from the GIs. We show using simulated channels that under a small number of taps our receiver pipeline can improve over the MMSE baseline and is close to MLSE which is the optimum receiver. We also test our receiver via experiments with an FPGA-based mm-wave testbed with phased antenna arrays and show that under a static scenario our receiver improves by $\sim$ 7 dB over the classic baseline MMSE-MAP using the channel estimation from the preamble and ~ 4 dB over the MMSE-MAP Oracle baseline. For future work we intend to extend our receiver pipeline to MIMO. In mobile scenarios, the gains are much higher with a throughput improvement of 620% over the baselines. Since obtaining channel estimates from the GIs that do not contain interference is not straightforward in MIMO scenarios, it will be highly interesting to explore what networks can learn to bootstrap the outdated channel information.

Acknowledgments

This paper is supported by the projects TSI-063000-2021-59 RISC-6G and TSI-063000-2021-63 MAP-6G funded by the Ministry of Economic Affairs and Digital Transformation under the European Union NextGeneration-EU plan, and by the Madrid Regional Government through the TAPIR-CM program (S2018/TCS4496).

REFERENCES

[1] S. Troia, R. Alvizu, Y. Zhou, G. Maier, and A. Pattavina, "Deep learning-based traffic prediction for network optimization," in *2018 20th ICTON*. IEEE, 2018, pp. 1–4.

[2] S. Zhang, S. Zhao, M. Yuan, J. Zeng, J. Yao, M. R. Lyu, and I. King, "Traffic prediction based power saving in cellular networks: A machine learning method," in *Proceedings of the 25th ACM SIGSPATIAL*, 2017, pp. 1–10.

[3] S. H. Haji and S. Y. Ameen, "Attack and anomaly detection in iot networks using machine learning techniques: A review," *Asian journal of research in computer science*, vol. 9, no. 2, pp. 30–46, 2021.

[4] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563–575, 2017.

[5] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep Learning Based Communication Over the Air," *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 132–143, 2017.

[6] J. Hoydis, F. A. Aoudia, A. Valcarce, and H. Viswanathan, "Toward a 6g ai-native air interface," *IEEE Communications Magazine*, vol. 59, no. 5, pp. 76–81, 2021.

[7] O. Shental and J. Hoydis, "Machine llrning: Learning to softly demodulate," in *2019 IEEE GC Wkshps*. IEEE, 2019, pp. 1–7.

[8] Z. Chang, Y. Wang, H. Li, and Z. Wang, "Complex cnn-based equalization for communication signal," in *IEEE 4th ICSIP*, 2019, pp. 513–517.

[9] D. Garcia Marti, D. Badini, D. De Donno, J. Widmer *et al.*, "Scalable machine learning algorithms to design massive mimo systems," in *ACM/IEEE MSWIM*, 2021.

[10] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, 2019.

[11] H. He, S. Jin, C.-K. Wen, F. Gao, G. Y. Li, and Z. Xu, "Model-driven deep learning for physical layer communications," *IEEE Wireless Communications*, vol. 26, no. 5, pp. 77–83, 2019.

[12] M. Goutay, F. A. Aoudia, J. Hoydis, and J.-M. Gorce, "Machine learning for mu-mimo receive processing in ofdm systems," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 8, pp. 2318–2332, 2021.

[13] C.-S. Choi, M. Piz, and E. Grass, *Non-Ideal Radio Frequency Front-End Models in 60GHz Systems*. John Wiley and Sons, Ltd, 2010, ch. 3.

[14] J. Vieira, F. Rusek, and F. Tufvesson, "Reciprocity Calibration Methods for Massive MIMO Based on Antenna Coupling," in *2014 IEEE GLOBECOM*. IEEE, 2014, pp. 3708–3712.

[15] S. Blandino, C. Desset, G. Mangraviti, A. Bourdoux, and S. Pollin, "Phase-Noise Mitigation at 60 GHz with a Novel Hybrid MIMO Architecture," in *Proceedings of the 2nd ACM Workshop on Millimeter Wave Networks and Sensing Systems*. New York, NY, USA: Association for Computing Machinery, 2018, p. 39–44.

[16] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2017.

[17] M. Honkala, D. Korpi, and J. M. Huttunen, "Deeprx: Fully convolutional deep learning receiver," *IEEE Transactions on Wireless Communications*, vol. 20, no. 6, pp. 3925–3940, 2021.

[18] D. Korpi, M. Honkala, J. M. Huttunen, and V. Starck, "Deeprx mimo: Convolutional mimo detection with learned multiplicative transformations," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–7.

[19] C. Han, A. O. Bicen, and I. F. Akyildiz, "Multi-wideband waveform design for distance-adaptive wireless communications in the terahertz band," *IEEE Transactions on Signal Processing*, vol. 64, no. 4, 2015.

[20] S. Tarboush, H. Sarieddeen, M.-S. Alouini, and T. Y. Al-Naffouri, "Single-versus multi-carrier terahertz-band communications: A comparative study," *arXiv preprint arXiv:2111.07398*, 2021.

[21] T. S. Rappaport, R. W. Heath Jr, R. C. Daniels, and J. N. Murdock, *Millimeter wave wireless communications*. Pearson Education, 2015.

[22] H.-W. Chan, C.-T. Wu, C.-W. Jen, C.-Y. Liu, W.-C. Lee, and S.-J. Jou, "A pseudo mmse linear equalizer for 60ghz single carrier baseband receiver," in *2017 IEEE 12th ASICON*, 2017, pp. 643–646.

[23] Standards Committee, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 3: Enhancements for Very High Throughput in the 60 GHz Band," *IEEE Std 802.11g-2003*, no. June, 2003.

[24] IEEE working group, "IEEE Draft Standard for Information Technology-Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks-Specific Requirements Part 11: Wireless LAN MAC and PHY Specifications-Amendment," *IEEE P802.11ay/D3.0*, pp. 1–763, March 2019.

[25] Measurement Campaigns and Initial Channel Models for Preferred Suitable Frequency Ranges. [Online]. Available: https://ec.europa.eu/research/participants/documents/downloadPublic?documentIds=080166e5a7a6b182\&appId=PPGMS

[26] J. O. Lacruz, R. Ruiz, and J. Widmer, "A real-time experimentation platform for sub-6 ghz and millimeter-wave MIMO systems," in *ACM MobiSys'21*, 2021.

[27] J. G. Proakis and M. Salehi, *Digital communications*. McGraw-hill New York, 2001, vol. 4.

[28] R. Steele and L. Hanzo, *Mobile radio communications: Second and third generation cellular and WATM systems*. IEEE Press-John Wiley, 1999.

[29] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[30] G. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference," *IEEE Transactions on Information theory*, vol. 18, no. 3, pp. 363–378, 1972.

[31] H. Vikalo, B. Hassibi, and U. Mitra, "Sphere-constrained ml detection for frequency-selective channels," *IEEE transactions on communications*, vol. 54, no. 7, pp. 1179–1183, 2006.

[32] IEEE Std 802.11, "Part 11: Wireless lan mac phy specifications," *IEEE Standard for Information technology, Telecommunications and information exchange between systems. Local and metropolitan area networks, Specific requirements*, 2020.

[33] IEEE Std 802.11ax, "Part 11: Wireless lan mac and phy specifications. amendment 1: Enhancements for high efficiency wlan." *IEEE Standard for Information technology, Telecommunications and information exchange between systems. Local and metropolitan area networks, Specific requirements*, 2021.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[35] W. Liu, F. Yeh, T. Wei, C. Chan, and S. Jou, "A Digital Golay-MPIC Time Domain Equalizer for SC/OFDM Dual-Modes at 60 GHz Band," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 10, pp. 2730–2739, Oct 2013.

[36] S. Jaeckel, L. Raschkowski, K. Börner, and L. Thiele, "Quadriga: A 3-d multi-cell channel model with time evolution for enabling virtual field trials," *IEEE transactions on antennas and propagation*, vol. 62, no. 6, 2014.

[37] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[38] IEEE Std 802.11, "Part 11: Wireless lan mac and phy specifications. ieee standard for information technology, telecommunications and information exchange between systems. local and metropolitan area networks,specific requirements," 2016.

[39] Sivers Semiconductors, *EVK06002 Development Kit*, 2022, https://www.sivers-semiconductors.com/sivers-wireless/evaluation-kits/.

[40] V. Va, H. Vikalo, and R. W. Heath, "Beam tracking for mobile millimeter wave communication systems," in *IEEE GlobalSIP*, 2016.

[41] Q. Qin, L. Gui, P. Cheng, and B. Gong, "Time-varying channel estimation for millimeter wave multiuser mimo systems," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 10, pp. 9435–9448, 2018.

[42] L. Cheng, G. Yue, D. Yu, Y. Liang, and S. Li, "Millimeter wave time-varying channel estimation via exploiting block-sparse and low-rank structures," *IEEE Access*, vol. 7, pp. 123 355–123 366, 2019.

[43] B. Azari, H. Cheng, N. Soltani, H. Li, Y. Li, M. Belgiovine, T. Imbiriba, S. D'Oro, T. Melodia, Y. Wang, P. Closas, K. Chowdhury, and D. Erdoğmuş, "Automated deep learning-based wide-band receiver," *Computer Networks*, vol. 218, p. 109367, 2022.

[44] P. Aggarwal, A. Gupta, and V. A. Bohara, "A guard interval assisted ofdm symbol-based channel estimation for rapid time-varying scenarios in ieee 802. lip," in *2015 IEEE 26th PIMRC*. IEEE, 2015, pp. 100–104.

[45] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Communications Letters*, vol. 23, no. 4, pp. 652–655, 2019.

[46] Q. Hu, F. Gao, H. Zhang, S. Jin, and G. Y. Li, "Deep learning for channel estimation: Interpretation, performance, and comparison," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, 2020.

[47] Y. Liao, Y. Hua, and Y. Cai, "Deep learning based channel estimation algorithm for fast time-varying mimo-ofdm systems," *IEEE Communications Letters*, vol. 24, no. 3, pp. 572–576, 2019.