Horizon Europe Framework Programme

Towards an AI-native, user-centric air interface for 6G networks

D3.5 Report on End-to-End Air-Interface Learning

Contractual Delivery Date:	30.06.2025
Actual Delivery Date:	25.06.2025
Editor: *(name, organization)*	Sebastian Cammerer, NVIDIA
Deliverable nature:	Report
Dissemination level:	Public
Version:	1.0

Keywords: End-to-end learning, air-interface learning, Constellations

ABSTRACT

This report explores the feasibility and practical relevance of end-to-end learning approaches for next generation 6G air interfaces. Our research focuses on three key innovations: pilotless communication systems, scalable symbol modulation learning, and joint source channel coding for short packet transmissions. The pilotless communication system eliminates traditional reference signals by jointly training a neural receiver with custom trainable constellations, embedding channel estimation mechanisms implicitly within the transmitted data. This approach demonstrates competitive block error rates while achieving up to 8% higher goodput compared to 5G NR baseline implementations due to a reduced piloting overhead. We further introduce a scalable autoencoder structure capable of supporting any M-ary modulation through a single AI/ML model, with robustness against non-linear impairments as expected in future high-frequency communications. For short packet transmissions, we propose joint source channel coding and modulation (JSCCM) mechanisms specifically optimized for compressed CSI feedback, addressing implementation challenges like an increased peak-to-average power ratio (PAPR) when multiplexing with other logical channels. These innovations require minimal modifications to existing infrastructure, providing a practical pathway toward intelligent, reconfigurable physical layers for future wireless systems while maintaining backward compatibility with current standards.

This project has received funding from the European Union's Smart Networks and Services Joint Undertaking (SNS JU) under grant agreement No. 101096379.

Disclaimer

This document contains material, which is the copyright of certain CENTRIC consortium parties, and may not be reproduced or copied without permission.

All CENTRIC consortium parties have agreed to full publication of this document.

Neither the CENTRIC consortium as a whole, nor a certain part of the CENTRIC consortium, warrant that the information contained in this document is capable of use, nor that use of the information is free from risk, accepting no liability for loss or damage suffered by any person using this information.

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101096379. This publication reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Impressum

Full project title: Towards an AI-native, user-centric air interface for 6G networks

Short project title: CENTRIC

Number and title of the work package: WP3 AI-AI Physical Layer Methods

Number and title of task: T3.2 End-to-End (E2E) Air-Interface Learning

Document title: D3.5 Report on End-to-end Air-Interface Learning

Editor: Sebastian Cammerer, NVIDIA

Work-package leader: Ramoni Ojekunle Adeogun, AAU

Copyright notice

Executive summary

The CENTRIC project aims to develop an AI-native, user-centric air interface for 6G networks, with this report focusing specifically on end-to-end air-interface learning approaches. As wireless networks evolve toward 6G, they face increasing demands for spectral efficiency, adaptability to emerging applications, and resilience to hardware limitations. Our research addresses these challenges through three interconnected innovations: pilotless communication systems, scalable symbol modulation learning, and joint source-channel coding for short packets.

Pilotless Communication Systems

A significant portion of our work demonstrates the feasibility of communication systems that function without traditional pilot signals. Current 5G NR implementations rely heavily on demodulation reference signals (DMRS) for channel estimation, which consume valuable spectrum resources. Our approach eliminates this overhead by jointly training a neural receiver capable of sophisticated signal processing and a custom trainable constellation that adapts to channel characteristics.

Through end-to-end learning, the system discovers optimal constellation configurations that embed channel estimation mechanisms implicitly within the transmitted data. We observe the emergence of "anchor symbols" that function as self-organized reference points, enabling accurate channel estimation without dedicated pilots. Performance evaluations reveal that this pilotless system not only matches conventional approaches in terms of block error rate but achieves up to 8% higher goodput due to reduced overhead.

The system also demonstrates robust performance when trained to compensate for hardware impairments like carrier frequency offset. Importantly, these gains can be realized with minimal modifications to existing 5G NR implementations, requiring only custom modulation tables and a non-piloting mode, both achievable through minor protocol extensions.

Scalable Symbol Modulation Learning

To address the challenges of supporting diverse modulation schemes, we propose a scalable autoencoder structure capable of handling any M-ary modulation through a single AI/ML model. This innovation eliminates the need to train separate networks for each modulation order, providing significant implementation advantages.

The system includes a neural network-based symbol modulator that converts encoded bits to complex symbols, a corresponding demodulator that maps received signals back to bits, and a flexible architecture supporting different constellation sizes through zero-padding. For practical deployment, we outline signaling mechanisms and training procedures, including a proposed Training Reference Signal (TR-RS) for online adaptation.

Our simulation results demonstrate that this approach not only matches the performance of traditional QAM in AWGN channels but also significantly outperforms conventional modulation in the presence of non-linear impairments such as phase noise—a critical advantage for high-frequency communications in the sub-THz bands.

Joint Source Channel Coding for Short Packets

The third innovation addresses the specific challenges of short packet transmissions, focusing on compressed Channel State Information (CSI) feedback. Rather than treating compression, channel coding, and modulation as separate processes, we propose a joint source channel coding and modulation (JSCCM) approach that integrates all three functions within a single end-to-end optimized system.

This approach eliminates inefficiencies from converting between different signal domains, optimizes the entire transmission chain for the specific characteristics of CSI data, and achieves higher compression efficiency with the same complexity. We specifically address practical implementation challenges, such as increased Peak-to-Average Power Ratio (PAPR) when non-QAM symbols are multiplexed with other logical channels. The report outlines three modes for CSI reporting and provides methods for resource allocation that minimize PAPR impact, ensuring compatibility with existing systems.

Conclusion and Future Directions

The innovations presented in this report demonstrate the potential of end-to-end learning approaches to transform wireless physical layer design. By jointly optimizing signal processing blocks that were traditionally designed in isolation, we achieve significant performance gains while maintaining practicality for real-world deployment.

As the wireless ecosystem moves toward cloud-native and AI-driven architectures, these approaches provide a foundation for a scalable, intelligent, and reconfigurable physical layer. Our work aligns with the CENTRIC vision of continuous adaptation even after deployment, allowing communication systems to evolve in response to new applications, hardware innovations, and regulatory changes. Future research will expand these concepts to multi-user scenarios, investigate online adaptation mechanisms, and contribute to standardization efforts for AI-native communication protocols in 6G and beyond.

List of authors

Company	Author	Contribution
NVIDIA	Sebastian Cammerer	Contributor, Editor
NVIDIA	Jakob Hoydis	Contributor
InterDigital	Ahmet Serdar Tan	Contributor
InterDigital	Anouar Yatribi	Contributor
AAU	Ramoni Adeogun	Reviewer, Co-Editor
SEQ	Efstathios Katranaras	Reviewer

Table of Contents

List of figures and tables

List of figures:

List of tables:

Abbreviations

Abbreviation	Full form
ADC	Analog-to-Digital Converter
ADAM	Adaptive Moment Estimation
AI	Artificial Intelligence
AWGN	Additive White Gaussian Noise
BCE	Binary Cross Entropy
BS	Base Station
CFO	Carrier Frequency Offset
CSI	Channel State Information
DMRS	Demodulation Reference Signal
JSCCM	Joint Source Channel Coding and Modulation
LLR	Log Likelihood Ratios
MCS	Modulation and Coding Scheme
MIMO	Multiple Input Multiple Output
NW	Network
NR	New Radio
OFDM	Orthogonal Frequency-Division Multiplexing
PARR	Peak-to-Average Power Ratio
PHY	Physical Layer
PUCCH	Physical Uplink Control Channel
PUSCH	Physical Uplink Shared Channel
QAM	Quadrature Amplitude Modulation
QPSK	Quadrature Phase Shift Keying
SGD	Stochastic Gradient Descent
SNR	Signal-to-Noise Ratio
SSCC	Separate Source Channel Coding
TDL	Tapped Delay Line
TQI	Training Quality Indicator
TR-RS	Training Reference Signal
UE	User Equipment
6G	Sixth Generation

1 Introduction

One of the core promises of the CENTRIC project is the development of communication systems that are capable of continuous adaptation even after deployment. Unlike conventional wireless systems that remain fixed in design for a decade or more, CENTRIC envisions future systems that can evolve post-deployment to meet the rapidly changing demands of emerging applications, hardware innovations, and spectrum regulations. This vision is particularly relevant given the accelerated pace of technological change. New use cases emerge on the timescale of months—often long after the rollout of a wireless standard. Similarly, semiconductor technology nodes and hardware capabilities advance continuously, creating a moving target for system design. Meanwhile, regulatory bodies are increasingly opening up new spectrum bands that could be exploited by flexible, software-defined radios. In such a dynamic environment, communication systems that can adapt their transmission behavior after deployment become essential for resilience, scalability, and long-term usability.

Another aspect of end-to-end (E2E) learning, is the capability to optimize systems with respect to an end-to-end performance metric. While classical systems are typically composed as chains of individually optimized signal processing blocks (such as the channel coding, demapper, or the equalizer), the E2E learning idea promises system optimization directly with respect to the final performance metric such as the overall block error rate.

Within this broader context, the work reported here explores end-to-end learning of the air interface—a novel approach that aligns with and extends the CENTRIC roadmap. While other efforts, particularly in Task 3.4, focused on neural receivers capable of replacing traditional signal processing chains of the receiver, this work under Task 3.2 goes further by introducing learning capabilities into the transmitter.

The key innovation is the joint training of a neural receiver and a trainable transmitter using custom constellations or even novel waveforms. We demonstrate that this technology allows to build systems which can communicate effectively without relying on traditional demodulation reference signals (DMRS). Unlike existing 5G NR implementations, which rely heavily on structured pilots for channel estimation, the proposed architecture removes the need for explicit pilot signals entirely. Instead, the system learns to embed channel estimation mechanisms implicitly into the transmitted data, guided by an end-to-end loss function. This becomes particularly important for systems where the piloting overhead can dominate the efficiency – such as for short message communications. Additionally, end-to-end learning can discover novel symbol constellations that are robust to hardware impairments, such as phase noise, which play an increasingly important role for communications in the sub-THz band.

The potential of our approach to future wireless networks is significant. By eliminating pilot overhead and jointly optimizing the signal design, such systems can achieve higher spectral efficiency, improved robustness to hardware impairments, and provide greater adaptability to specific environments. These qualities are central to the CENTRIC mission of enabling intelligent, future-proof wireless networks, and they point toward a viable path for next-generation standards such as 6G and beyond.

2 End-to-end learning for the Air-interface

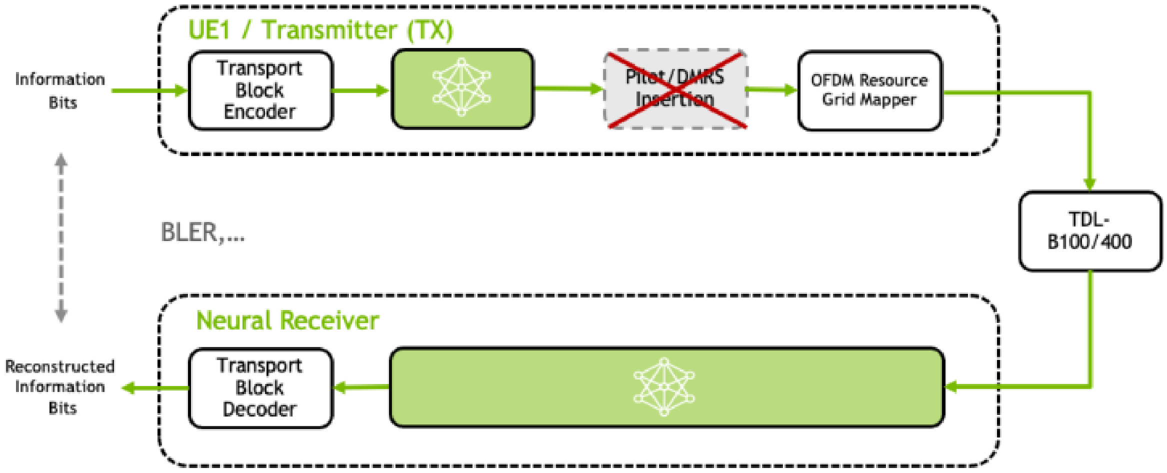


Figure 1: ***End-to-end learning of a pilotless communication scheme by extending the Neural receiver (NRX) with a trainable custom constellation***

This section focuses on the development and evaluation of pilotless communication systems enabled by end-to-end (E2E) learning, a technique that represents a significant departure from classical 5G NR physical layer designs. Rather than relying on predefined signal processing blocks and dedicated reference signals, E2E learning treats the entire transceiver chain—including both transmitter and receiver—as a trainable system, optimized jointly to maximize performance under real-world channel conditions. In doing so, it offers a promising pathway towards more efficient, adaptive, and intelligent wireless communication architectures.

In 5G NR terminology, the system depends on the demodulation reference signal (DMRS) for channel estimation. While effective, DMRS consume valuable time-frequency resources resulting in an effective rate loss of the overall system. In this work, we eliminate the need for explicit pilots altogether by allowing the receiver to infer channel state information directly from the structure of the transmitted signal. This is made possible through a neural receiver (NRX) trained alongside a trainable custom constellation at the transmitter, enabling reliable data reconstruction without any pilot overhead.

Unlike conventional systems that rely on fixed QAM modulation, our approach treats the constellation points as learnable parameters, allowing the transmitter to discover modulation schemes that are tailored to the channel characteristics and receiver behavior. These custom constellations often evolve to include symbol structures that implicitly support channel estimation—a phenomenon we refer to as *the emergence of anchor symbols*. These symbols act as self-organized references within the data stream, demonstrating the system's ability to internalize roles typically reserved for explicit pilots.

The results presented in this section highlight the feasibility of this approach, demonstrating not only competitive block error rate (BLER) performance but also substantial gains in goodput, thanks to the reduced overhead. The architecture is based on and extends the NRX developed in Task 3.4, now operating in a fully pilotless mode with support for joint transmitter-receiver optimization. While this methodology departs from 5G compliance, it remains close to implementable reality through minimal modifications—such as pilot masking and constellation table updates—and provides a compelling blueprint for future 6G systems.

All code and experiments of this section are available as open source implementation on Github.

2.1 System Model

We restrict the work on pilotless communications to single user MIMO OFDM communication system where a user with N_{TX} transmit antennas sends messages to a receiver equipped with N_{RX} receive antennas. Such a communication system is illustrated in Figure 1.

We assume an OFDM-based system with N_S and N_F being the number of OFDM symbols and subcarriers forming the resource grid (RG), respectively. The following description is in the frequency-domain, i.e., assuming post-FFT samples and a sufficiently long cyclic prefix (CP).

For reliable communications, the transmitter encodes a bit vector $\mathbf{b}_{\mathrm{n_F,n_S,}} \in \{0,1\}^{\mathrm{m}}$ of m bits on every resource element (RE) $[n_F, n_S]$ allocated for data transmission, where $1 \leq \mathrm{n_F} \leq \mathrm{N_F}$ and $1 \leq n_S \leq N_S$. To that aim, every vector $\mathbf{b}_{\mathrm{n_F,n_S}}$ is mapped onto a complex-valued baseband symbol denoted by $\mathrm{x}_{\mathrm{n_F,n_S}} \in \mathbb{C}$ which is typically done by using a 2^m quadrature amplitude modulation (QAM) with Gray labelling. In the following, we will introduce custom constellations, i.e., $\mathrm{x}_{\mathrm{n_F,n_S}}$ is not on the regular QAM grid anymore.

This leads to the OFDM RG of baseband modulated symbols

$$\mathbf{X} = \begin{bmatrix} x_{1,1} & \cdots & x_{1,N_S} \\ \cdots & & \cdots \\ x_{N_F,1} & \cdots & x_{N_F,N_S} \end{bmatrix}.$$

In 5G NR, some positions in $\mathbf{X}$ are not used for data transmission but send known pilots, so-called demodulation reference signal (DMRS) pilots. This is explicitly not done in the following experiment, i.e., all entries of $\mathbf{X}$ carry data symbols.

Hence, the received RG is denoted by as $\mathbf{Y} = \{\mathbf{y}_{\mathrm{n_F,n_S}}\}_{1\leq \mathrm{n_F} \leq \mathrm{N_F}, 1 \leq \mathrm{n_S} \leq \mathrm{N_S}}$ where $\mathbf{y}_{\mathrm{n_F,n_S}} \in \mathbb{C}^{\mathrm{N_{RX}}}$ is the received signal for the RE $[n_F, n_S]$ and given as

$$\boldsymbol{y}_{n_F,n_S} = \boldsymbol{H}_{n_F,n_S}\boldsymbol{x}_{n_F,n_S} + \boldsymbol{w}_{n_F,n_S}$$

Where $\boldsymbol{x}_{n_F,n_S} = \left[x_{n_F,n_S,1}, \dots, x_{n_F,n_S,N_{TX}}\right]$ is the vector of transmitted baseband symbols, $\boldsymbol{H}_{n_F,n_S} \in \mathbb{C}^{N_{\mathrm{RX}} \times N_{\mathrm{TX}}}$ is the channel matrix, and $\boldsymbol{w}_{n_F,n_S} \sim \mathcal{CN}\left(\mathbf{0}, \sigma^2 \boldsymbol{I}_{d_{N_{\mathrm{RX}}}}\right)$ is the complex-valued additive white Gaussian noise (AWGN) with noise power σ^2. Additional effects such as hardware impairments could be easily integrated in this model.

The task of the receiver is now to produce an estimate of the transmitted bit sequence $\hat{\mathbf{b}}_{\mathrm{n_F,n_S}} \in \{0,1\}^{\mathrm{m}}$ or log likelihood-ratios (LLRs) thereof. Let $\ell_{\mathrm{n_F,n_S,i}} \in \mathbb{R}$ denote the LLR associated to the *i*-th bit of $\hat{\mathbf{b}}_{\mathrm{n_F,n_S}}$.

As channel model, we consider the 3GPP 38.901 TDL models, in particular, we use the TDL-C model with 100 Hz Doppler shift and 300ns delay spread. It is worth noting that this channel model is fully differentiable. During training, we fix the SNR in a first stage when learning the custom constellations and finetune the receiver in a second step on a randomized SNR to further optimize the robustness of the system.

For further details on the system model, we refer the reader to previous deliverables D3.2 [7] and D3.3 [8].

2.2 End-to-end Learning & Custom Constellations

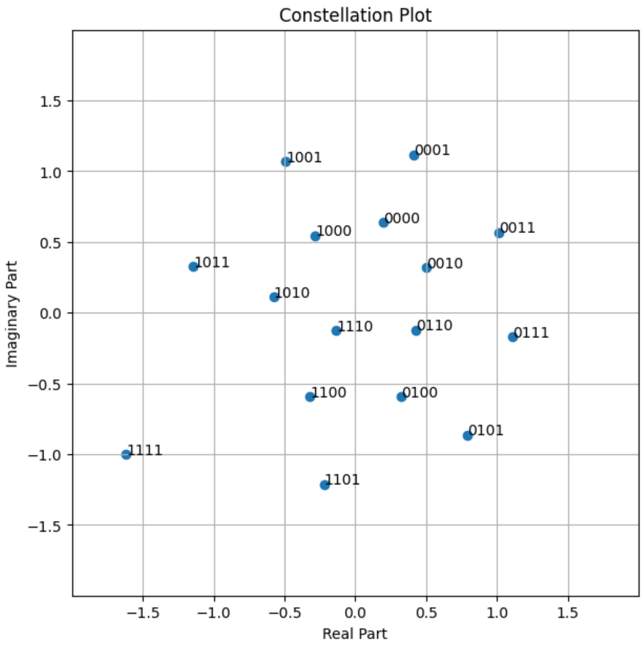


Figure 2 ***Learned custom constellation for pilotless communications of 16 symbols, i.e., each symbol transmits 4 bits.***

Our system consists of two major components: a trainable transmitter that employs custom constellation points $x_{n_F,n_S} \in \mathbb{C}$ instead of standard modulation, and a neural network-based receiver that learns to decode signals and estimate the channel without relying on pilot symbols [1]. These components are trained jointly in an end-to-end manner using a differentiable channel model. As such, the concept requires two key enablers which will be explained in the following:

1) Custom constellations

2) DMRS-free slot configuration

The end-to-end system (including the channel model) is fully differentiable [4] and, hence, SGD-based training is straightforward. We use the ADAM optimizer with learning rate $l = 10^{-3}$, and the BCE loss function. The task of the receiver is to solve $N_F \times N_S \times m$ binary classification problems in parallel from the received signal $\mathbf{Y}$. We can estimate the BCE by Monte-Carlo integration:

$$\mathcal{L} \approx -\frac{1}{BN_{\mathrm{F}}N_{\mathrm{S}}m}\sum_{b=0}^{B-1}\sum_{n_F=1}^{N_{\mathrm{F}}}\sum_{n_S=1}^{N_{\mathrm{S}}}\sum_{i=0}^{m-1}\Big[b^{[b]}_{n_F,n_S,i}\log(\,\sigma(\ell_{n_F,n_S,i}(\boldsymbol{Y}^{[b]}))) + (1-b^{[b]}_{n_F,n_S,i}(\log(\,\sigma(-\ell_{n_F,n_S,i}(\boldsymbol{Y}^{[b]}))))\Big]$$

where B is the batch size, the superscript *[b]* is used to refer to the b^{th} batch example, and σ(⋅) is the logistic sigmoid function. $\ell_{n_F,n_S,i}(\boldsymbol{Y})$ denotes the LLR computed by the detector from the received signal $\boldsymbol{Y}$ for the i^{th} bit of layer n_{T} transmitted over the RE $[n_F, n_S]$. As LLRs are binary logits, $\sigma(\ell_{n_F,n_S,i}(\boldsymbol{Y}))$ gives the corresponding probability for the bit to be equal to one, given $\boldsymbol{Y}$. For training, we average the loss over all iterations to ensure that inference of the receiver can be done for a flexible number of iterations.

As described in D3.6, an additional loss on the channel estimate can be used to further improve the training convergence.

For further details on the NRX architecture, we refer to deliverable D3.3 and the open-source code release in D3.4 available via https://github.com/NVlabs/neural_rx. One difference is that we do not provide least squares (LS) channel estimates as input of the neural receiver since no DMRS pilots are available for the initial LS channel estimation.

Note that the custom constellation points $\mathrm{x}_{\mathrm{n_F,n_S}} \in \mathbb{C}$ of the transmitter are now also trainable parameters of the network, i.e., gradients of the loss w.r.t. $\mathrm{x}_{\mathrm{n_F,n_S}}$ must be calculated. As the end-to-end system is implemented in NVIDIA Sionna, this is done via TensorFlow's automated gradient computation. Note that the same custom constellations are used for all $1 \leq \mathrm{n_F} \leq \mathrm{N_F}$ and $1 \leq n_S \leq N_{\mathrm{S}}$ transmitted symbols. For $m = 4$, this means only 16 additional complex-valued, i.e., 32 real-valued weights must be trained while the neural receiver itself has >100k trainable parameters.

The transmitter starts with a classical (i.e., as specified in the 5G NR standard) 16-QAM constellation as initialization. During training, however, these constellation points are allowed to move within the complex plane as they are treated as trainable parameters. Over the course of training, the system learns to organize the constellation in a way that best supports data recovery at the receiver, given the characteristics of the channel and the absence of pilot symbols. An example of such a constellation after training is shown in Figure 2.

2.3 Interpretation of Pilotless Communications

Explaining the exact behavior of custom constellations in combination with neural receivers is challenging due to the black-box nature of neural networks. However, we have set up a simple experiment to gain an intuition of how the inherent channel estimation works.

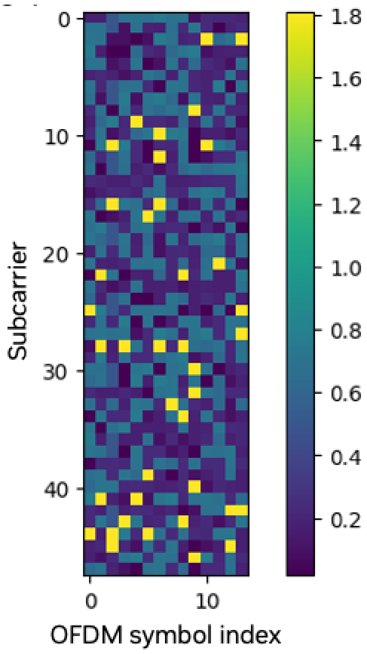


Figure 3: ***Relative average power of a received slot using custom constellations and 48 subcarriers (y-axis) and 14 OFDM symbols (x-axis).***

Our hypothesis is that one symbol acts as an "anchor" for channel estimation, standing out from the remaining symbols as shown in Figure 3. The other constellation points follow the expected behavior of non-uniform geometric shaping. It is important to note that the occurrence of such a strong symbol is reproducible, although the labeling is mostly random and depends on the specific training seed, among other factors. This phenomenon also strongly depends on the training SNR and does not occur if channel estimates are provided. Intuitively, gradient descent finds the best tradeoff between maximizing the information rate of the constellation and the required piloting overhead for a specific system configuration.

Since we have trained the NRX with *double_readout* functionality (see report D3.6), meaning the NRX returns channel estimates, we can visualize the result of channel estimation in Figure 4. Keep in mind that the NRX has not seen any classical pilot for channel estimation in this setup. In the following experiment, we keep the transmitted slot fixed but average over many channel realizations.

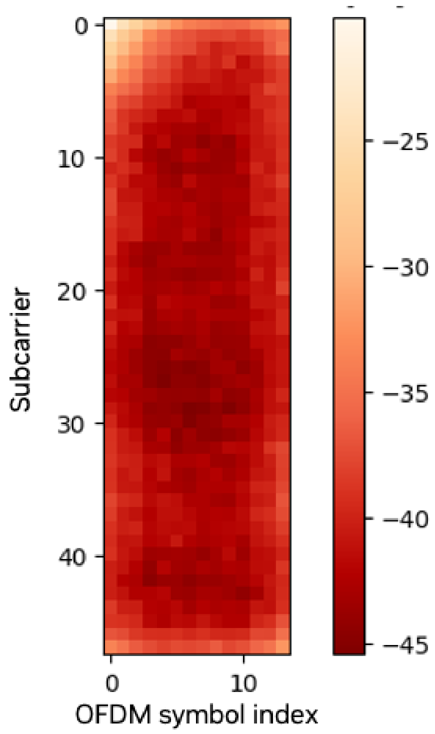


Figure 4: ***MSE Channel estimation error in dB for a received slot using custom constellations and 48 subcarriers (y-axis) and 14 OFDM symbols (x-axis).***

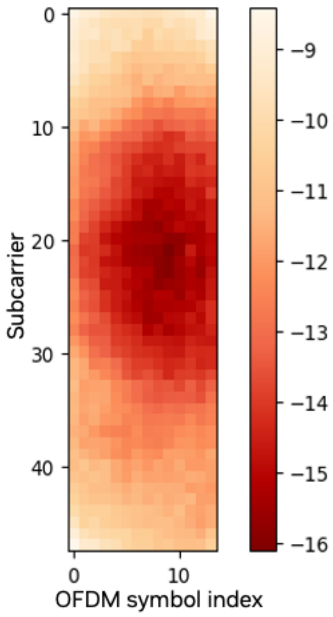


Figure 5: ***MSE channel estimation error in dB for a received slot using custom constellations and 48 subcarriers (y-axis) and 14 OFDM symbols (x-axis). This is simulated after removing the anchor symbol.***

As expected, in Figure 5 we observe a significantly larger channel estimation error when excluding the *anchor* symbol from the transmit message, i.e., we randomly select the transmitted symbols from the remaining 15 out of 16 constellation points. Figure 6 completes this comparison by evaluating MSE when excluding each of the custom constellation symbols from the transmit message.

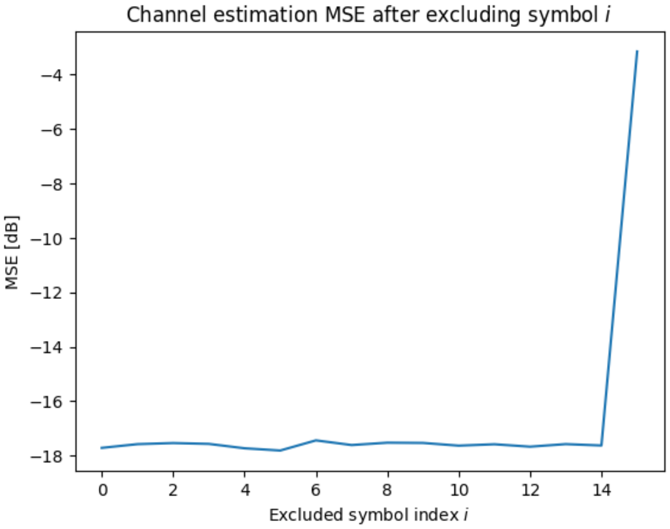


Figure 6: ***Channel estimation error after excluding the symbol with index i from the transmitter alphabet.***

As expected, we observe a significantly larger channel estimation error when excluding the *anchor* symbol (index $i = 15$) compared to excluding any other symbol. We conclude that this *anchor* symbol has a specific contribution to the implicit channel estimation carried out by the neural receiver.

2.4 Performance Evaluation

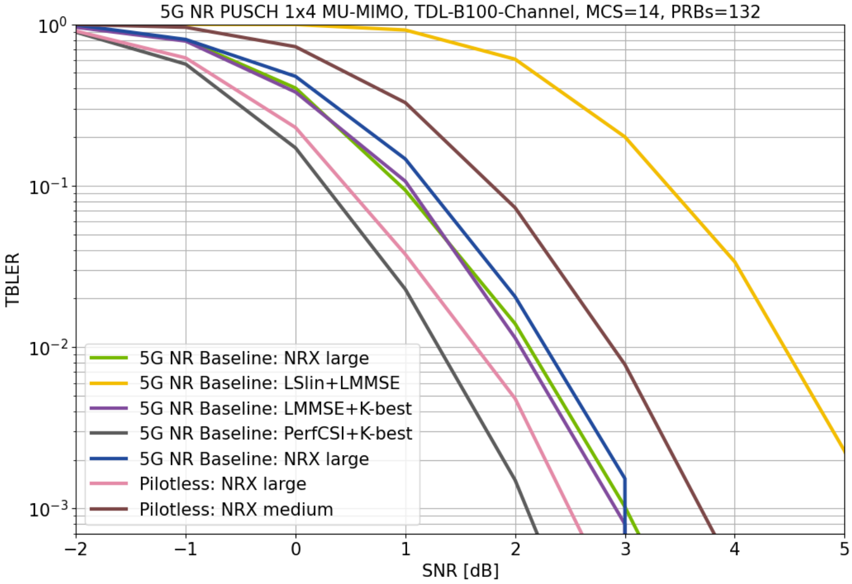


Figure 7: ***Block error rate performance evaluation on transport block level of the pilotless communications scheme in comparison with multiple baselines.***

We conducted a comprehensive evaluation of the trained system under realistic channel models. Initially, the system was trained using a TDL-C channel model and subsequently evaluated with a TDL-B model to assess generalization capabilities and avoid overfitting. This evaluation included comparisons against several standard 5G NR baselines, such as systems with least-square estimation followed by LMMSE filtering, systems with K-best detection, and systems with perfect channel knowledge. The results are shown in Figure 7.

The baselines are:

- LS channel estimation with linear interpolation between pilots and LMMSE-based MIMO detection
- LMMSE-based channel estimation with K-best MIMO detection. The LMMSE channel covariance matrix is generated for the 3GPP UMi channel (see D3.2 and D3.3) and K=64.
- Perfect channel state information (CSI) and K-best detection (K=64)

As can be seen, the NRX slightly outperforms the LMMSE+K-best baseline while benefiting from a lower computational complexity. Even the low complexity and real-time version operates less than 1 dB away from the aforementioned baseline.

In terms of block error rate (BLER), the pilotless system matched or outperformed the baseline configurations across a wide range of signal-to-noise ratios. More significantly, when measuring goodput—defined as the number of successfully transmitted information bits per resource element—the pilotless system demonstrated a consistent advantage. This advantage stems from the reduced pilot overhead and the system's ability to maintain performance without explicit channel knowledge.

Goodput measurements confirm that the neural receiver architecture can reliably decode information under moderate and high SNR conditions, with throughput improvements of up to 8% compared to the best-performing baseline systems. These gains are particularly relevant for high data rate applications and dense deployment scenarios, where spectral efficiency is critical.

For a fair comparison, we evaluate the goodput G of the system using the formula

$$G = \frac{(1 - P_{\mathrm{BLER}})N_{\mathrm{Payload}}}{N_{\mathrm{REs}}}$$

where P_{BLER} is the block error rate, N_{Payload} is the number of transmitted payload bits, and N_{REs} the number of occupied resource elements (including DMRS if used). The results are shown in Figure 8. As can be seen, the advantages of the pilotless scheme are more obvious when looking at the goodput instead of the BLER figures. This can be intuitively explained by the fact that the BLER results in Figure 7 compare systems of different information rates, i.e., the pilotless schemes transmit more information bits with the same occupied resources. This advantage does not show off in the BLER result, but becomes obvious when looking at the goodput result.

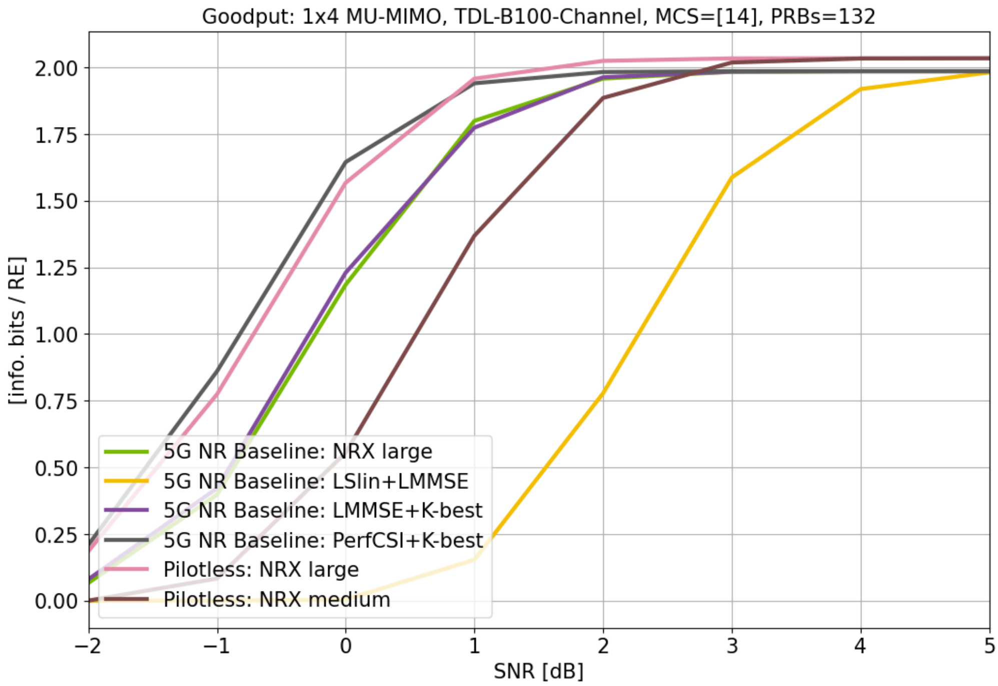


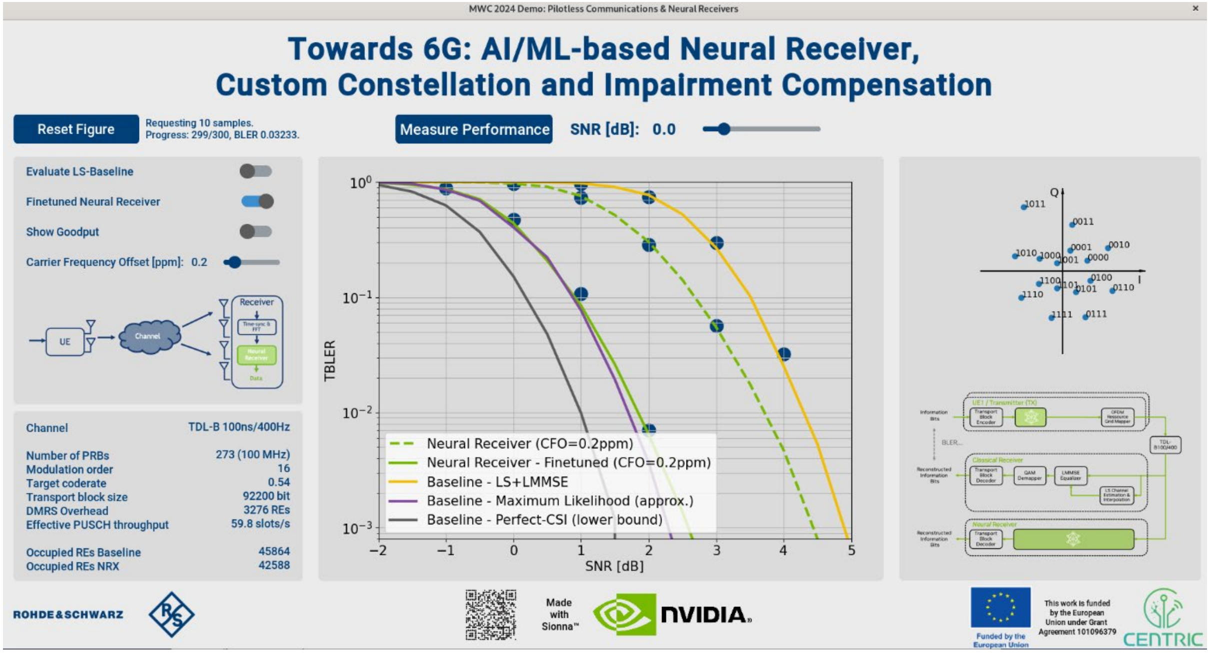
Figure 8: ***Goodput performance evaluation of the pilotless communications scheme in comparison with multiple baselines.***

We want to emphasize that, in particular for short messages, the piloting overhead becomes large and degrades the overall system spectral efficiency significantly. Thus, pilotless communications is a promising approach to reduce the piloting overhead for short message communications.

2.5 Robustness and Training for Impairments

In practical systems, performance robustness in the presence of hardware impairments is crucial. To address this, we extended the training framework to simulate carrier frequency offset (CFO) during both training and evaluation. By sampling CFO values within a fixed range during training, the receiver learned to compensate for frequency mismatches introduced by hardware imperfections or Doppler shifts.

The results demonstrate that when trained accordingly, the system maintains its error performance even under significant CFO conditions. This highlights the flexibility of the neural receiver and the feasibility of integrating compensation for various impairments directly into the training process, thereby reducing the need for dedicated signal processing blocks in deployment.



MWC 2024 Demo: Pilotless Communications & Neural Receivers

Towards 6G: AI/ML-based Neural Receiver,
Custom Constellation and Impairment Compensation

Requesting 10 samples.
Progress: 299/300, BLER 0.03233.

SNR [dB]: 0.0

Evaluate LS-Baseline
Finetuned Neural Receiver
Show Goodput
Carrier Frequency Offset [ppm]: 0.2

Channel	TDL-B 100ns/400Hz
Number of PRBs	273 (100 MHz)
Modulation order	16
Target coderate	0.54
Transport block size	92200 bit
DMRS Overhead	3276 REs
Effective PUSCH throughput	59.8 slots/s
Occupied REs Baseline	45864
Occupied REs NRX	42588

This work is funded by the European Union under Grant Agreement 101096379

Figure 9: ***Screenshot from the Mobile World Congress 2024 Demonstrator together with our partner Rohde & Schwarz.***

We have demonstrated the impact of CFO in a hardware-in-the-loop testbed at Mobile World Congress 2024 together with Rohde & Schwarz. A screenshot of the live demo is shown in Figure 9. As can be seen, the CFO causes severe performance degradation of the pilotless communications scheme when not considered in the training. However, when included in the training data, the end-to-end system learns to compensate for the CFO. This is an important aspect for various use-cases with high CFO, either as a result from cheap hardware components, or as the result of high carrier frequencies such as the case in THz communications.

2.6 Summary

This work demonstrates the feasibility and practical relevance of pilotless communication systems enabled through end-to-end learning. By jointly training a neural receiver and a custom trainable constellation, we eliminate the need for explicit pilots such as DMRS, while maintaining reliable data reconstruction and achieving higher spectral efficiency. The system effectively learns signal structures that embed implicit reference mechanisms, enabling accurate channel estimation and equalization without relying on standardized pilot signals.

An important observation is that these gains can be realized with minimal modifications to existing 5G NR implementations. The approach requires (a) the use of custom modulation tables—already supported to some extent in the standard—and (b) a non-piloting mode, which would require only minor extensions to signaling protocols. This compatibility allows for rapid experimentation and potential incremental deployment without disrupting existing infrastructure.

As the wireless ecosystem moves toward open, cloud-native, and AI-driven architectures, the ability to adapt receiver behavior via training, rather than redesign, becomes increasingly valuable. The presented pilotless communication scheme is an early but powerful example of how learning-based transceiver design can unlock new capabilities for 6G systems, including increased data rates, reduced overhead, and environment-aware operation.

In line with the CENTRIC vision, this work lays the foundation for scalable, intelligent, and reconfigurable physical layers, providing a strong basis for future research in multi-user learning, online adaptation, and standardization of AI-native communication protocols.

3 End-to-end learning for non-codebook-based Symbol Modulations and Short Packets

This section focuses on the end-to-end learning schemes under two studies; (i) non-codebook based symbol modulations and (ii) short packets. The study on end-to-end learning for non-codebook-based symbol modulations aims to achieve scalable symbol modulation learning to support any M-ary modulation through a single AI/ML model. The study further includes practical aspects such as required signalling for online training. The other study on end-to-end learning for short packets focuses on the enablers and mechanisms to support joint source channel coding and modulation (JSCCM) for compressed CSI feedback, one of the use cases for short packet transmissions. The study defines methods to determine parameters and configurations for the JSCCM based compressed CSI feedback aiming to reduce the impact of impairments such as PARP.

3.1 Scalable Symbol Modulation Learning

3.1.1 Introduction

Symbol modulation and symbol demodulation are among the fundamental blocks of the PHY layer of wireless communications, as shown in Figure 10. Symbol modulators convert a group of encoded bits to complex symbols that represent the in-phase and quadrature components of the baseband signal, whereas symbol demodulators convert the received baseband complex signals to group of bits that are fed into the channel decoder. The number of bits carried within a symbol depends on the modulation order Q_m of the modulation scheme.

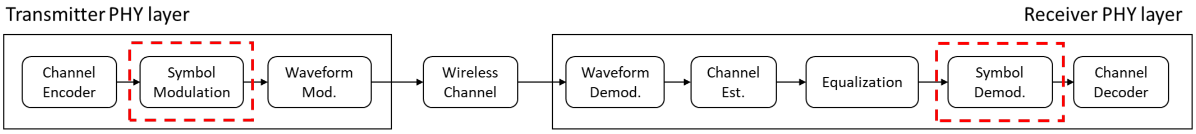


Figure 10: Main blocks in the PHY Layer

Modulation schemes used in 5G NR PDSCH are QAM-based and ranging from QPSK to 1024-QAM [9]. The modulation schemes are used in combination with the channel coding to determine the spectral efficiency of data transmission. The combination of modulation and coding schemes (MSC) in 5G NR is in Tables 5.1.3.1-1 to 5.1.3.1-4 in [10]. Prior to the start of downlink/uplink transmission, BS instructs the UE to select an MCS index table.

Autoencoders are a special type of unsupervised neural networks, wherein an autoencoder tries to find a low-dimensional representation (i.e., compressed) of the input at an intermediate layer that is reconstructed at the output with minimum error. In an autoencoder, the goal of the training is minimizing a cost function dependent on the difference/distance between desired output and actual output with respect to the neural network parameters. Autoencoders are typically used in dimensionality reduction, denoising, anomaly detection, data compression, etc. The goal of autoencoders in PHY layer is to find representations of input messages in an intermediate layer that are robust against the distortions created by wireless channel, transmitter/receiver hardware impairments, and any other effect that could impact an efficient transmission of a signal from the transmitter to the receiver.

In this study, we focus on autoencoder based symbol modulation where we propose and evaluate a scalable autoencoder structure supporting any M-ary symbol modulation and further discuss practical aspects such as signalling and online training.

3.1.2 System Model and Proposed Method

System Model

Autoencoder based symbol modulation learning is an end-to-end learning method which replaces the symbol modulation and demodulation blocks at the transmitter and receiver with trainable blocks as shown in Figure 11. Here, input to the autoencoder is the bits from the channel encoder output and output of the autoencoder is the probability of output being equal to input for the received symbol, when binary cross entropy is used as the loss function. The output probability at the output of autoencoder can be used to generate LLR values required for the channel decoder.

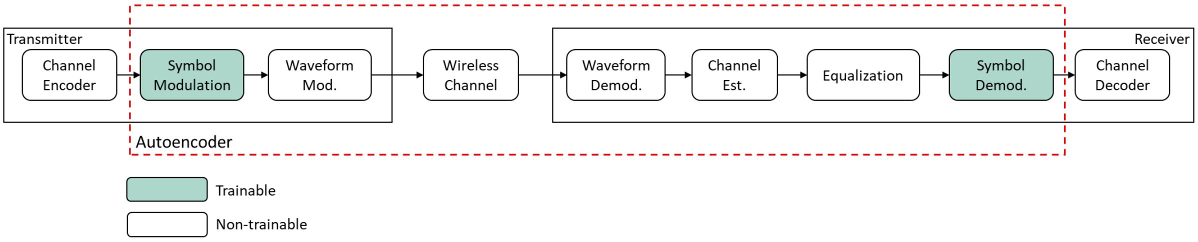


Figure 11*: End-to-end trainable symbol modulation*

One of the issues with autoencoder based symbol modulation is that separate autoencoders need to be trained for each modulation order as the size of input bits to autoencoder changes for each modulation order. In this study, we propose a scalable autoencoder structure that can be trained for M-ary modulations where M, i.e. size of the constellation can be changed arbitrarily. An exemplary structure is given in Figure 12. The inputs to the autoencoder are represented with +1 for binary 1 or -1 for binary 0, so that $x_i \in \{+1, -1\}$. Hence, for each M-ary symbol modulation there are $M = 2^m$ possible inputs to the autoencoder. The neural network output maps m inputs to two real valued outputs, y_I and y_Q, that are applied power normalization to ensure the same average bit energy (for bits in D) for every M-ary modulation. The outputs y_I and y_Q represent the in-phase and quadrature components of a complex symbol that is fed into the waveform modulator, such as OFDM. The scalable structure can be constructed for a maximum number of input size m^* while some of the inputs at transmitter side can be padded with zeros, so that $x_i = 0$ for $i \geq m$. The symbol demodulation at the receiver starts with the output samples of the equalizer, y_I' and y_Q', which are real valued samples that represent the in-phase and quadrature components of the received complex symbols. The samples are input to the demodulator NN at the receiver that has m outputs, $x_0', x_1', \dots, x_{m-1}'$, that represent the received encoded binary symbols where $x_i' \in [-1, +1]$. The autoencoder outputs x_i' at the receiver side for $i \geq m$ are discarded during the demodulation stage. The scalable NN structure allows maximum M^*-ary modulation where $M^* = 2^{m^*}$. The maximum number of input size m^* depends on the implementation complexity, training overhead, etc.

The scalable NN structure for modulation learning allows training for all possible M-ary modulations within a given single NN structure. Another option for NN structure for modulation learning is to construct separate NNs for each m, which is expected to incur additional complexity and memory requirements.

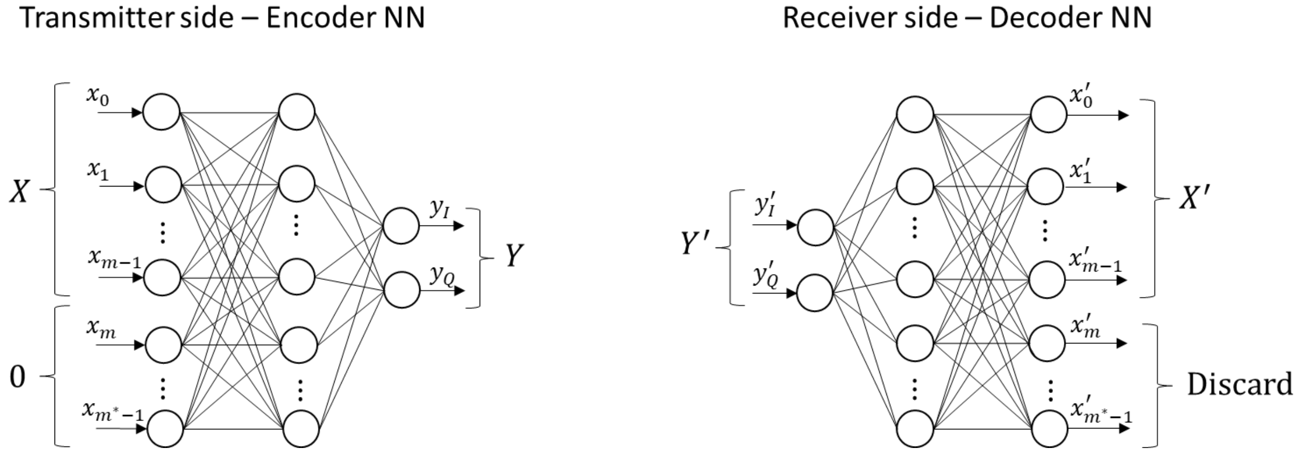


Figure 12: ***Scalable NN structure for M-ary symbol modulation, $M = 2^m$.***

The overall structure of the end-to-end symbol modulation learning in the PHY layer chain is further illustrated in Figure 13. Some of the blocks in between encoder and decoder of autoencoder are waveform modulation, transmitter RF, channel, receiver RF, waveform demodulation, equalization.

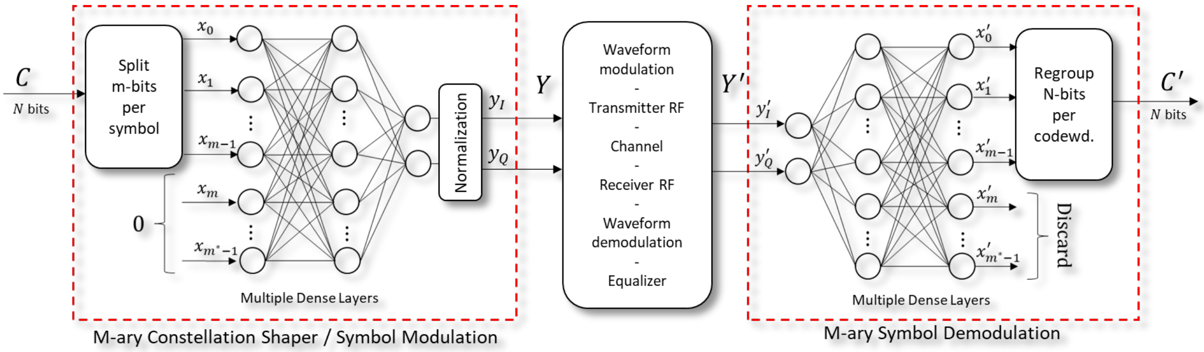


Figure 13: ***Scalable symbol modulation learning in PHY layer***

Training and Signalling

For the online training of the autoencoder based symbol modulation, a new type of reference signal may be defined, e.g., a training reference signal (TR-RS). The Training Reference Signals (TR-RS) are used as part of the feedforward computations to train a NN. A TR-RS consists of a single complex symbol that represents the output of the transmitter autoencoder NN (symbol modulator), i.e., $Y = y_I + y_Q j$, for a given input $X = \{x_0, \dots, x_{m-1}\}$, and on the contrary to current 5G NR modulation methods, y_I and y_Q are continuous real valued so that $\{y_I, y_Q \in \mathbb{R}: -1 < y_I < 1, -1 < y_Q < 1\}$ and quantized with the default precision prior to waveform modulation. The complex signals transmitted within a TR-RS are specific to the UEs that is determined based on the weights of the transmitter autoencoder NN. A TR-RS can be located on any resource element that are reserved to the UE following the initial access procedures. The sequence of X used to generate TR-RS is known to transmitter and receiver in advance. A typical sequence of X for modulation training includes all possible inputs of an M-ary modulation. For example, for $m = 4$, $X^0 = 0000, X^1 = 0001, \dots, X^{M-1} = 1111$, where X^i denotes the i-th input sequence. The complex symbol carried by the i-th TR-RS, $Y^i = y_I^i + y_Q^i j$, is computed based on the weights of the transmitter autoencoder NN.

Training process of the autoencoder NNs can be performed offline or online. Offline training can be performed within a computer simulator for a given channel statistic. However, offline training may not reflect the channel conditions specific to a UE. Offline trained NN weights can be used as initialization for online training. The weights and structure (number of layers, nodes), i.e., parameters, of the offline trained NN for different channels constitute a codebook for NN parameters that is shared between BS and UEs. After a UE completes the initial access procedures, BS selects parameters to be used for the corresponding BS-UE connection and sends the index of parameters from the shared codebook to the UE. An example shared codebook structure is given in Table 1 where the codebook table can include NN index, structure, weights, and the corresponding type of channel the NN is trained for. NN index denotes the index of the chosen NN parameters. Structure field is composed of an array of size of the number of layers and each element of the array indicates the nodes per layer. As an example, [2,16,8] structure denotes a NN with three layers and each layer has 2, 16, and 8 nodes, respectively. Weights field is composed of an array of size of the number of NN weights. As an example, [2,16,8] structure denotes a NN weight array of size $2 \times 16 + 16 \times 8 = 160$. In the table, structure, weights and type of channels are exemplary, and the table can be extended for any type of channel.

Table 1*: An example codebook of NN parameters for $m^* = 8$*

NN Index *I_{NN}*	Structure *(nodes per layer)*	Weights	Type of Channel
0	[2,8,8]	[0.0235, -0.4511, 0.7890, …]$_{1\times80}$	Stationary, high SNR
1	[2,8,8,8]	[0.3501, 0.0511, -0.2084, …]$_{1\times144}$	Low mobility, medium SNR
2	[2,16,8]	[-0.7394, -0.0034, 0.6318, …]$_{1\times160}$	Medium mobility, medium SNR
…	…	…	…

The flowchart and procedure for the proposed symbol modulation learning are given in Figure 14 and Figure 15 that includes joint online training at both BS and UE. The procedure steps are explained below.

Step-1: UE completes initial access procedures and sends UE specific information such as location, statistics of the underlying wireless channel and device impairment statistics (e.g., carrier frequency offset, ADC loss, timing offset, noise figure, etc.) to the BS.

Step-2: BS selects the index from the codebook of NN parameters, e.g., Table 1, based on the UE specific information and sends the index to the UE.

Step-3: UE initializes the NN model using the received index of the codebook and replies to BS with an initialization complete message via PUCCH.

Step-4: After BS receives the initialization complete message from the UE, BS sends a known TR-RS sequence to the UE to test the initialized NN.

Step-5: UE demodulates the output samples of the equalizer for the received TR-RS symbols to compute the receiver NN output loss. UE sends the training loss as part of the training quality indicator (TQI) which can include loss to BS within the CSI reporting feedback.

Step-6: BS receives the TQI.

- If the loss value in the TQI is above a threshold, then BS decides to retrain the symbol modulator/demodulator. BS sends Training Configuration Message to the UE to configure the training process with convergence and maximum iteration constraints. Go to Step-7.

- If the loss value in TQI is below a threshold, then BS decides there is no need for retraining. Then, the inference procedure starts.

Step-7: UE replies to BS with Training Ready message via PUCCH.

Step-8: BS sends training symbols via TR-RS to the UE.

Step-9: UE trains the receiver NN using the received TR-RS and updates the NN parameters by computing the receiver NN output loss and error values for the nodes in the NN.

- In case the loss converges within a maximum number of iterations, i.e., successful completion of the training, Go to Step-11.

- In case, the loss does not converge within a maximum number of iterations, i.e., unsuccessful training, UE sends Training Failure message to BS via PUCCH to terminate the training process. Upon receiving this message BS deploys conventional MCS for the symbol modulation.

- In case maximum number of iterations is not reached yet UE feedback the error values of the input nodes of the receiver NN to BS via Training Status Message. UE can also send the current loss and convergence within TQI periodically to the BS. Go to Step-10.

Step-10: BS trains the transmitter NN using the error values feedback from the UE.

- In case BS decides to terminate the training process, due to insufficient resources, power consumption, latency, loss and convergence values in TQI etc., UE and BS deploy conventional MCS for the symbol modulation.

- In case BS decides to stop the training, UE and BS deploys current trained parameters.

- In case BS decides to continue training process, go to Step-8.

Step-11: UE sends Training Complete message to BS via PUCCH.

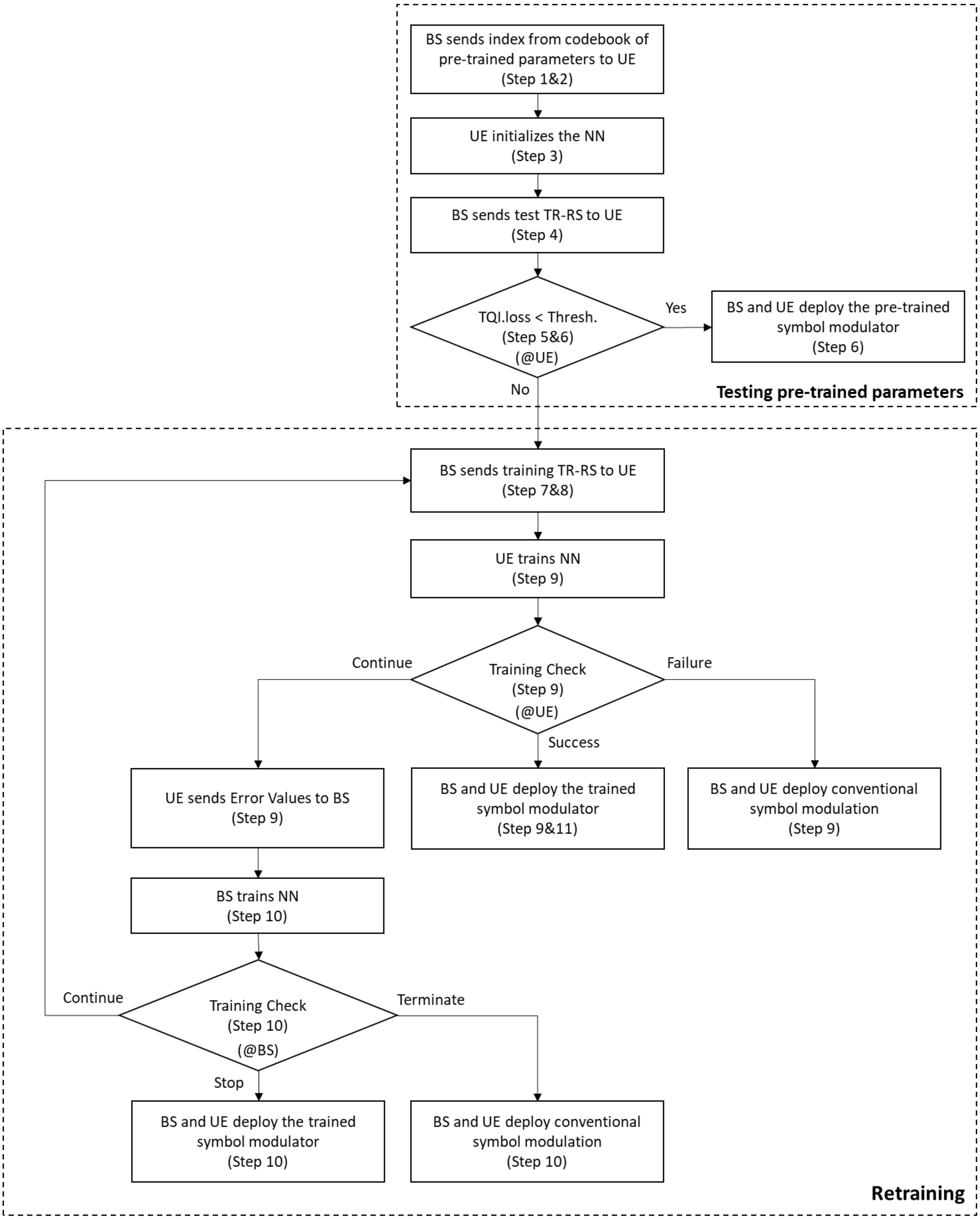

BS sends index from codebook of pre-trained parameters to UE (Step 1&2)
UE initializes the NN (Step 3)
BS sends test TR-RS to UE (Step 4)
TQI.loss < Thresh. (Step 5&6) (@UE)
Yes
BS and UE deploy the pre-trained symbol modulator (Step 6)
No
Testing pre-trained parameters
BS sends training TR-RS to UE (Step 7&8)
UE trains NN (Step 9)
Continue
Training Check (Step 9) (@UE)
Failure
Success
UE sends Error Values to BS (Step 9)
BS and UE deploy the trained symbol modulator (Step 9&11)
BS and UE deploy conventional symbol modulation (Step 9)
BS trains NN (Step 10)
Continue
Training Check (Step 10) (@BS)
Terminate
Stop
BS and UE deploy the trained symbol modulator (Step 10)
BS and UE deploy conventional symbol modulation (Step 10)
Retraining

Figure 14*: Flowchart of the symbol modulation training*

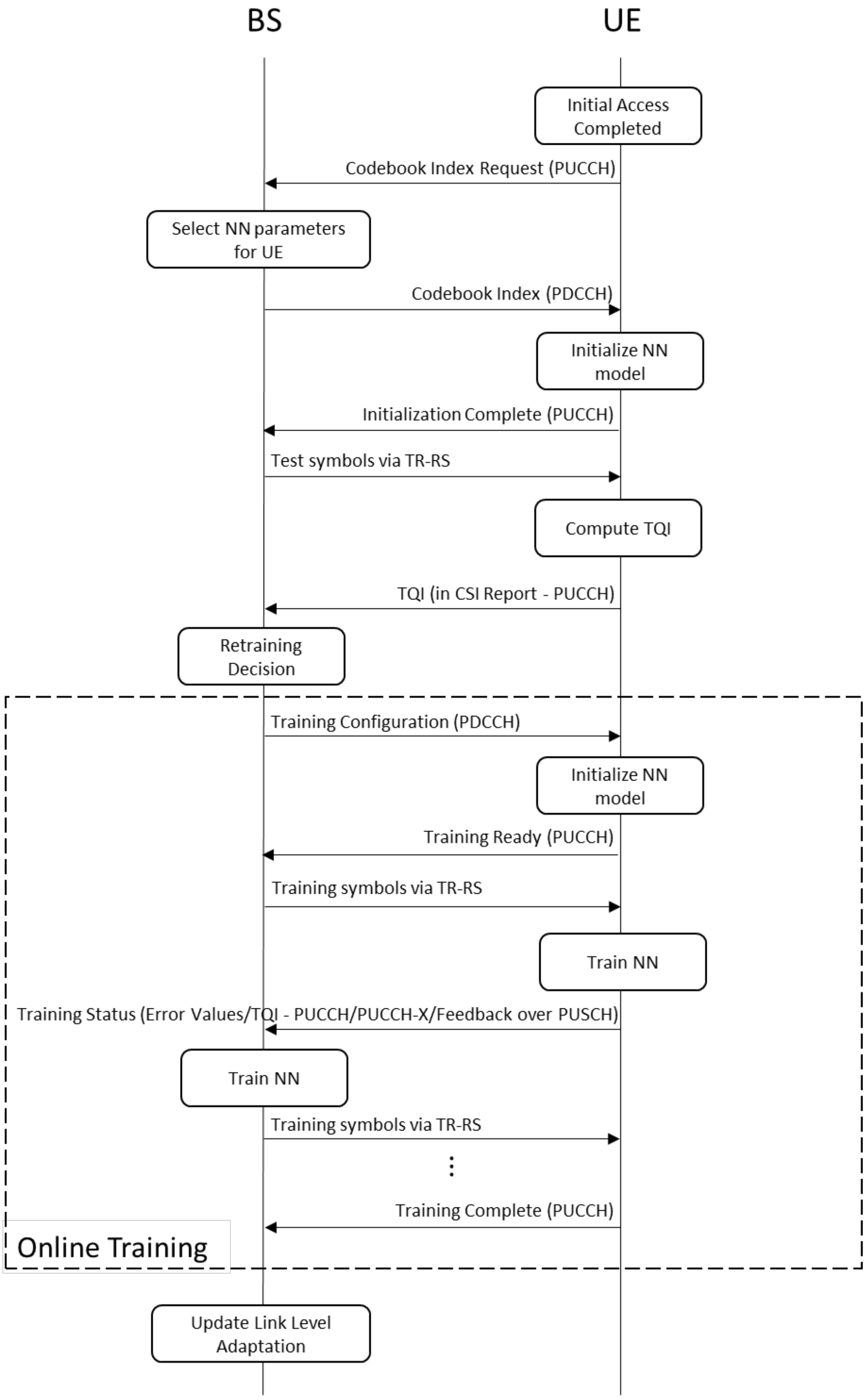

BS
UE
Initial Access Completed
Codebook Index Request (PUCCH)
Select NN parameters for UE
Codebook Index (PDCCH)
Initialize NN model
Initialization Complete (PUCCH)
Test symbols via TR-RS
Compute TQI
TQI (in CSI Report - PUCCH)
Retraining Decision
Training Configuration (PDCCH)
Initialize NN model
Training Ready (PUCCH)
Training symbols via TR-RS
Train NN
Training Status (Error Values/TQI - PUCCH/PUCCH-X/Feedback over PUSCH)
Train NN
Training symbols via TR-RS
⋮
Training Complete (PUCCH)
Online Training
Update Link Level Adaptation

Figure 15*: Procedure for symbol modulation learning - online training at BS and UE*

3.1.3 Performance Evaluation

In this section, we evaluate the performance of the proposed scalable symbol modulation learning method through simulations. First, we evaluate the performance under additive white Gaussian noise (AWGN) and then under an additional non-linear impairment, phase noise, especially prevalent in high frequency communications.

The effective channel between Y and Y' can be modelled with any differentiable channel model and used for the training of the autoencoder model. Assuming that the channel model is AWGN, so that $Y' = Y + n$, where $n \sim N(0, P_n)$ and $m^* = 6$, the evaluation results of the trained model are provided in Figure 16 and Figure 17. In Figure 16, the learned constellations and received symbols are provided. In Figure 17, the symbol error rate (SER) for the QAM-based and autoencoder-based symbol modulation is provided, where the results are similar showing that the autoencoder-based method can learn the optimal constellation shape in AWGN.

Another set of results are provided in Figure 18 and Figure 19 for a channel with AWGN and phase noise, $Y' = Y \cdot e^{-j\psi} + n$, where $\psi \sim N(0, \alpha\|x\|^2)$ and $n \sim N(0, P_n)$. In Figure 18, the learned constellations and received symbols are provided, where the learned constellations are no longer QAM-like due to non-linear phase noise. In Figure 19, the SER for the QAM-based and autoencoder-based symbol modulation is provided for AWGN and phase noise. The results of symbol modulation learning show improved SER compared to traditional QAM-based modulation in the presence of non-linear phase noise.

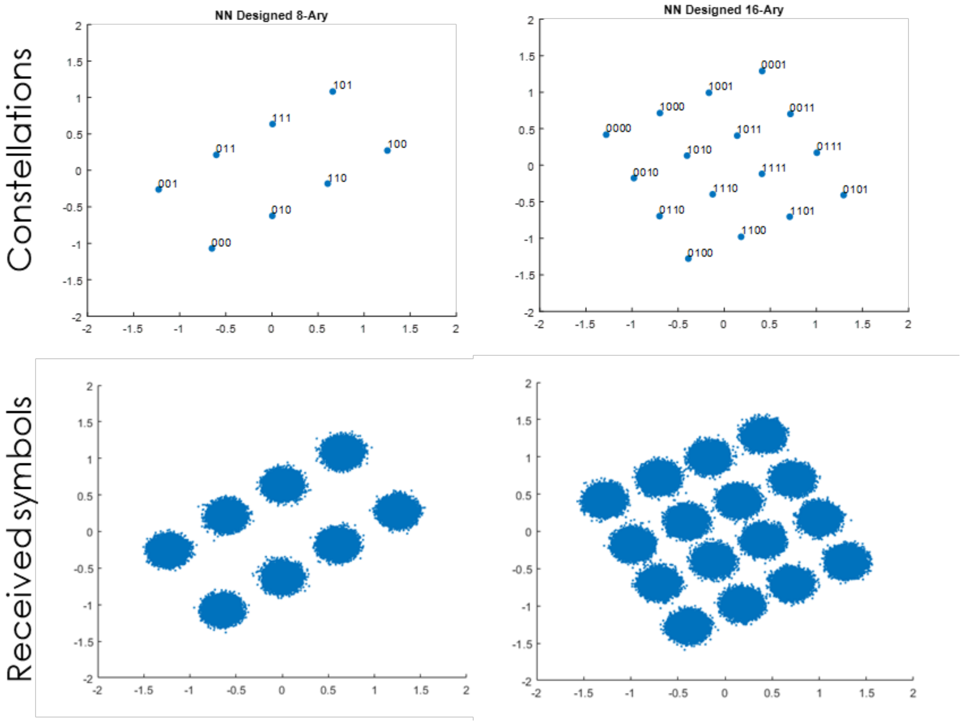


Figure 16*: NN-designed constellations for AWGN channel*

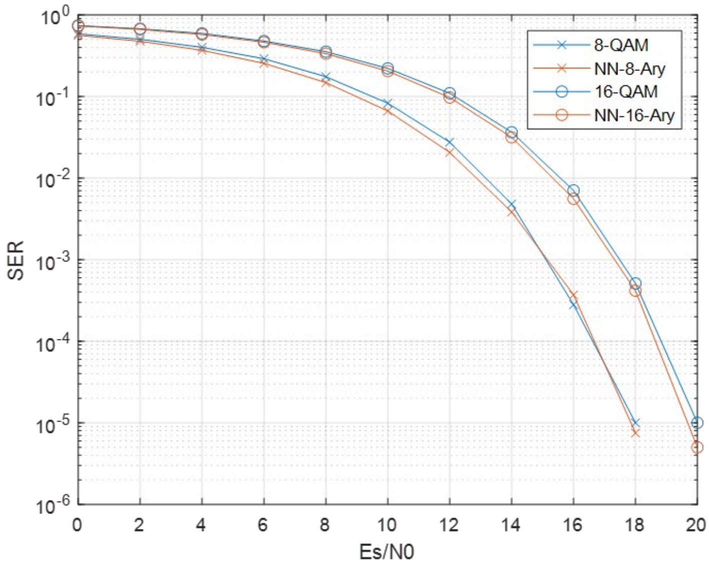


Figure 17*: SER vs Es/N0 for AWGN channel*

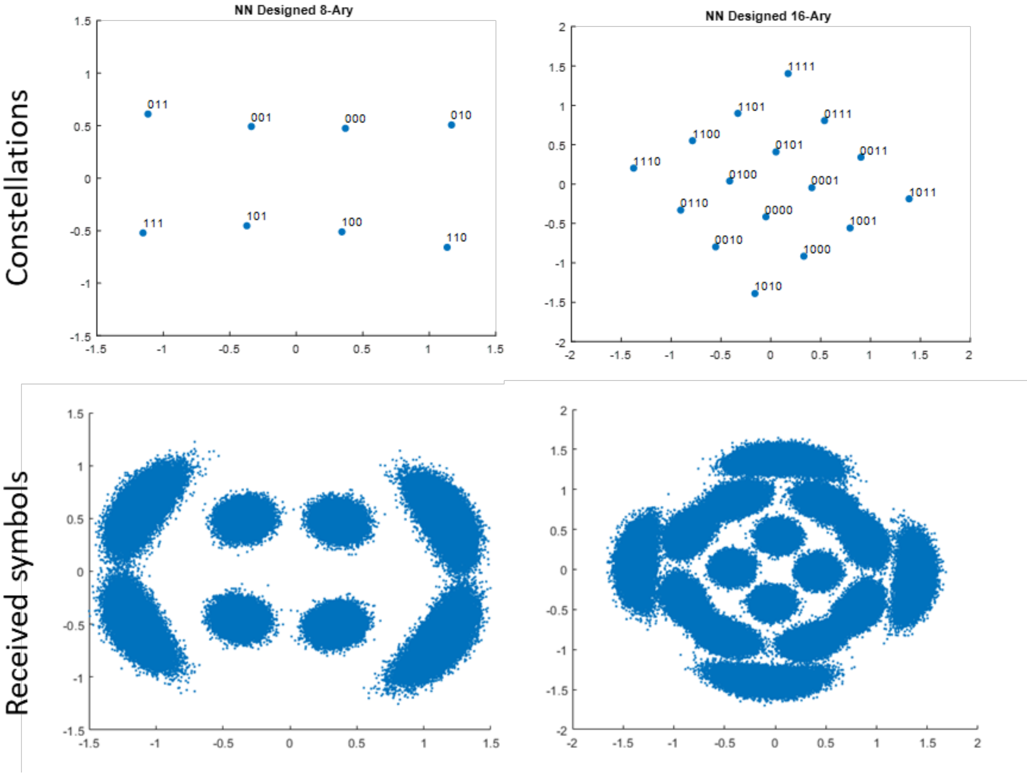


Figure 18*: AE-designed constellations for AWGN channel and phase noise*

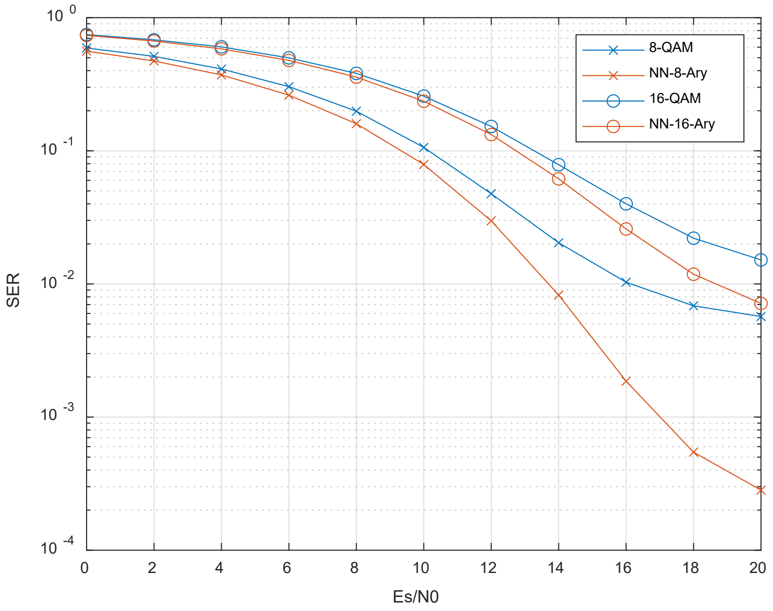


Figure 19*: SER vs Es/N0 for AWGN channel and phase noise*

3.1.4 Conclusion

In this study, we have proposed a scalable autoencoder structure to support any M-ary modulation for symbol modulation learning. In the proposed method, the symbol modulation and demodulation blocks are replaced with neural network blocks where the input and output of the autoencoder are zero-padded during training and inference according to the selected M-ary modulation. We have evaluated the performance of the proposed methods via simulations and demonstrated the learned constellations and performance improvement under different channel conditions including non-linear phase noise that is prevalent in high frequencies. We have further investigated required signalling and mechanisms to enable symbol modulation learning for beyond 5G and 6G communications.

3.2 Joint Source Channel Coding and Modulation for Short Packet Transmissions

3.2.1 Introduction

End-to-end learning for short packets has various use cases in wireless communications. One of these use cases, AI/ML based CSI compression, is studied in 3GPP as a means to reduce the UL CSI feedback reporting overhead while acquiring higher resolution DL channel state information at the NW. The size of the CSI feedback is significantly smaller than the data packets, around an order of magnitude. AI/ML-based CSI compression uses an end-to-end two-sided autoencoder (AE) model, where the encoder part is located at the transmitter (UE) side and compresses the high dimensionality input data (e.g., CSI) to a lower dimensionality latent vector; the decoder part is located at the receiver (NW) side and performs the reconstruction based on the received latent vector (e.g., compressed CSI).

Existing approaches to compress the Channel State Information (CSI) focus only on the compression of CSI where the compression efficiency is limited due to subsequent legacy blocks of channel encoder and symbol modulation. Given that the size of CSI data is small compared to data packets, the subsequent legacy blocks, i.e., channel encoder/decoder and symbol modulator/demodulator can be part of the end-to-end learning process. In this study, we focus on the enablers for efficient mechanisms for the reporting of compressed CSI through an end-to-end learning based joint source channel coding and modulation mechanism.

3.2.2 System Model and Proposed Method

From an end-to-end perspective, current studies in 3GPP on CSI compression focus on the sperate source channel coding (SSCC) approach wherein the compressed CSI is channel coded and symbol modulated separately, as illustrated in Figure 22. In SSCC approach, the output of the encoder of CSI compression is quantized and converted to binary to be used as input for the channel coding. Joint source channel compression and modulation (JSCCM) approach is a more efficient end-to-end compression alternative for CSI feedback wherein the Autoencoder (AE) is trained to perform CSI compression, channel coding and symbol modulation jointly, as illustrated in Figure 23. In JSCCM approach, the output of the encoder of the CSI compression is complex valued and can be used as input to IFFT block (i.e., OFDM Mapping and Modulation).

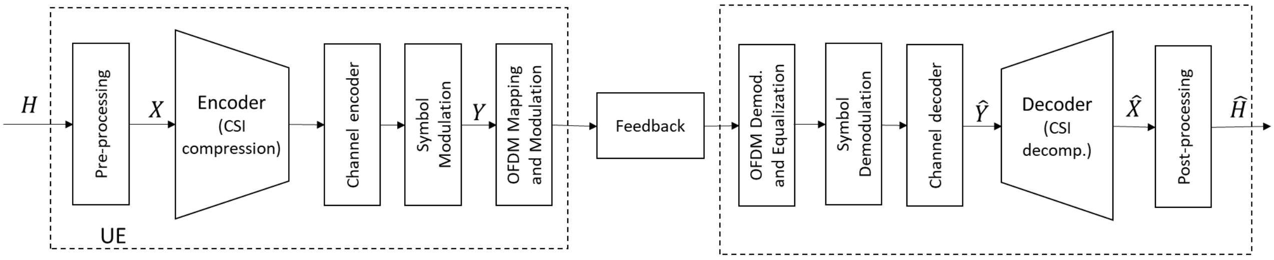


Figure 20*: Example SSCC end-to-end structure*

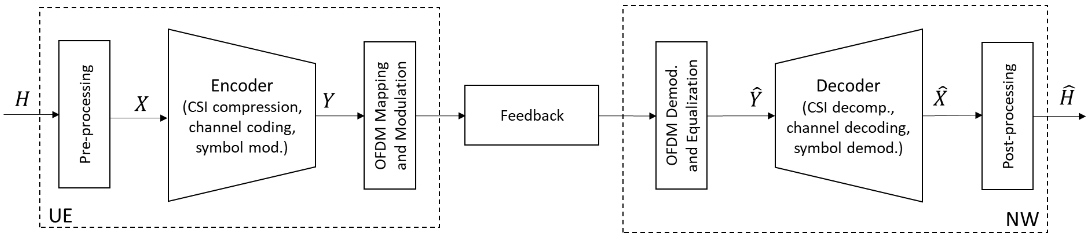


Figure 21*: Example JSCCM end-to-end structure*

As opposed to SSCC method, where the input to the OFDM block constitutes QAM-based discrete valued complex symbols, in the JSCCM method, the input to ODFM block is non-QAM based and continuous valued. Existing OFDM modulation techniques are typically engineered for QAM-based symbols to handle impairments such as peak-to-average-signal-ratio (PAPR) accordingly. Non-QAM based continuous valued compressed CSI symbols (CCS) at the output of the encoder of autoencoder may cause increased PARP.

The increase in PAPR may cause performance degradation in data channels and control channels depending on the selected mode of reporting the CCS. We define three different modes of reporting the CCS via legacy logical channels, i.e., PUSCH and PUCCH, as shown in Figure 24. The first mode, Mode-1, assumes that the CCS are multiplexed onto resources of a scheduled PUSCH transmission. The second mode, Mode-2, assumes that the CCS are explicitly transmitted over a configurable and/or indicated PUCCH resource. The third mode, Mode-3, assumes that the UE may use a dedicated uplink channel for CCS transmission. For example, the NW may configure a CCS channel which is only dedicated for CCS transmission for Mode-3. A user equipment may determine the mode through a configuration from the Network or it may determine and report the mode for CCS reporting.

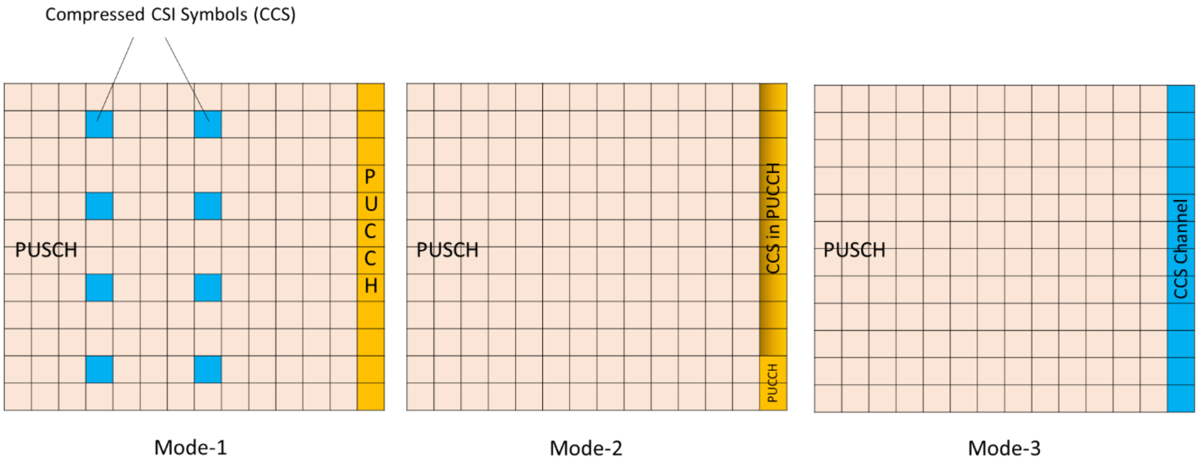


Figure 22: Modes of CCS Reporting

In case the CCS reporting will be done through Mode-1, the UE may further determine a pattern for the CCS, as shown in Figure 25. The UE may select the best CCS pattern out of available patterns that minimizes the PAPR when the CCS is multiplexed with PUSCH symbols. In addition, when CCS reporting mode is Mode-1, there may be two options for the multiplexing of the CCS symbols with PUSCH symbols, where in the first option CCS symbols are overwritten on PUSCH symbols and in the second option additional resources for CCS are allocated.

In the first option, the CCS symbols may replace (i.e., overwrite) the corresponding PUSCH symbols in the OFDM grid. In this option, the MCS of the PUSCH may be relaxed (i.e., lower index MCS) to support successful decoding of the overwritten PUSCH symbols. For the selection of the CCS resources (e.g., locations of CCS resources in Figure 25), one method is determining the resources to minimize the final PARP when the puncturing resources in PUSCH are replaced with CCS symbols. For this method, the UE may be configured with a set of patterns for CCS resources, and the UE may determine the index of CCS pattern that results in lowest PAPR in the OFDM grid of multiplexed CCS and PUSCH. The UE may compute PAPR_1, the average PARP for PUSCH multiplexed with CCS pattern#1, and PAPR_2, the average PAPR for PUSCH multiplexed with CCS pattern#2 and then chose the pattern with lowest PARP. For the selection of the CCS resources, another method is determining CCS resources such that the distance between the PUSCH symbols in the selected pattern and the CCS symbols is minimized.

In the second option, additional PUSCH resources could be provided so that no PUSCH symbols are overwritten on the OFDM grid. This option would not distort the throughput or error performance of PUSCH but increases the overhead. Similar to the previous option, the CCS resources can be determined such that the final PAPR is minimized when CCS are multiplexed with PUSCH resources.

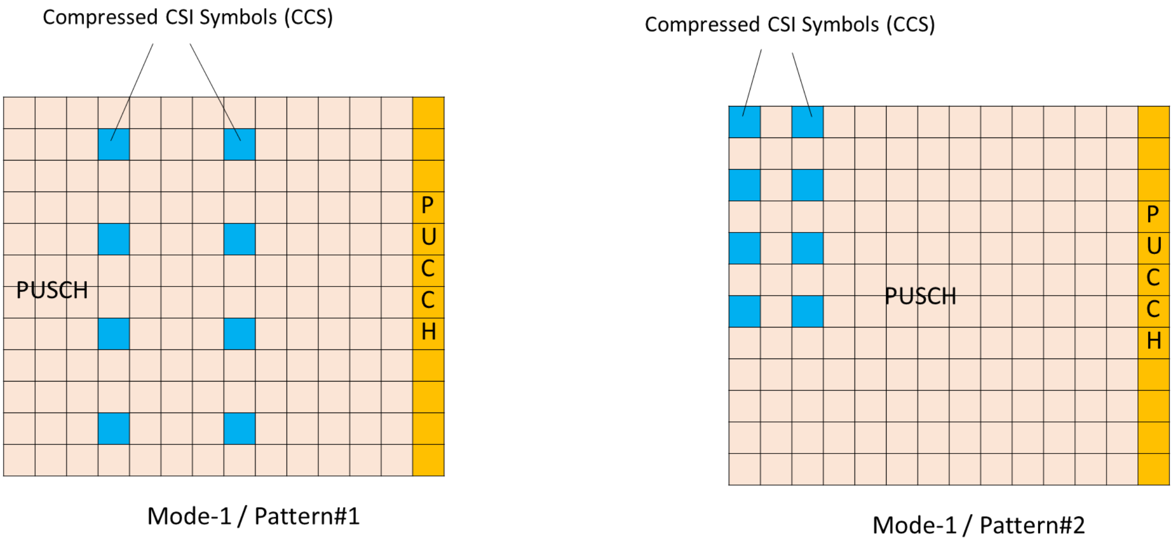


Figure 23: Example CCS Patterns for Mode-1 Reporting

3.2.3 Expected Outcomes

The methods described in this study focus on improving the efficiency of an end-to-end learning based short packet transmission use case, i.e., CSI compression. In the existing discussions in 3GPP, CSI compression is based on separate source channel coding (SSCC), i.e., compressed CSI is first channel coded and then modulated. The performance of this scheme is limited due to fixed blocks of channel encoder and symbol modulation where the compressed CSI symbols are converted to bits first for channel coding and then converted back to symbol domain via symbol modulation.

End-to-end learning based on joint course channel coding and modulation (JSCCM) aims at removing the bottlenecks caused by conversion between different blocks and quantized symbol modulation. The JSCCM approach may lead to improved KPIs, i.e., Squared Generalized Cosine Similarity (SGCS) and throughput, with same complexity and autoencoder model. Thanks to the relatively short packet sizes of CSI feedback compared to data packets, the model complexity could be kept limited to achieve practical implementations at a UE.

3.2.4 Conclusion

In this study, we have proposed mechanisms to enable joint source channel coding and modulation (JSCCM) for CSI compression, a potential use case for end-to-end learning for short packet transmissions. Different to the existing schemes for CSI compression, the JSCCM scheme further includes channel coding and symbol modulation within the trainable blocks. We have proposed new methods to handle the impairments, such as increased PAPR, that would be caused by the transmission of non-QAM based symbols especially when they multiplexed with other logical channels, e.g., PUSCH. In this way, the potential performance benefits of JSCCM based CSI feedback would not be limited by the impairments in practical implementation in a 6G communication system.

4 Conclusions

This report demonstrates that end-to-end learning fundamentally redefines physical layer design by enabling joint optimization of traditionally isolated communication blocks, achieving significant gains in spectral efficiency (8% higher goodput), hardware impairment tolerance, and protocol flexibility. The proposed pilotless systems, scalable modulation architectures, and integrated JSCCM approaches provide concrete pathways for deploying AI-native air interfaces within existing infrastructure, balancing innovation with practical implementation constraints. These technological breakthroughs could position 6G not merely as an evolutionary upgrade but as a transformative platform where neural networks dynamically adapt to emerging hardware capabilities, regulatory requirements, and application demands—ushering in an era where communication systems continuously self-optimize through embedded intelligence.

References

[1] F. Aït Aoudia and J. Hoydis, "End-to-end Learning for OFDM: From Neural Receivers to Pilotless Communication," IEEE Trans on Wireless Commun., 2021.

[2] D. Korpi, M. Honkala, and J. Huttunen, "Deep Learning-Based Pilotless Spatial Multiplexing," arXiv preprint arXiv:2312.05158, 2023.

[3] 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, Dec. 2017.

[4] 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, 2017.

[5] F. Aït Aoudia and J. Hoydis, "Model-free Training of End-to-end Communication Systems," IEEE Journal on Selected Areas in Communications, vol. 37, no. 11, 2019.

[6] S. Cammerer, F. Aït Aoudia, S. Dörner, M. Stark, J. Hoydis, and S. ten Brink, "Trainable Communication Systems: Concepts and Prototype," IEEE Transactions on Communications, vol. 68, no. 9, 2020.

[7] D3.2 ML Algorithms for GPU Implementation, CENTRIC Project Deliverable

[8] D3.3 Concept Delivery for PoC Implementation in WP5, CENTRIC Project Deliverable

[9] 3GPP TS 38.211, NR; Physical channels and modulation, Technical Specification, v18.6.0.

[10] 3GPP TS 38.214, NR; Physical layer procedures for data, Technical Specification, v18.6.0.