A Study of Neural Polar Decoders for Communication

Rom Hirsch
Ben-Gurion University
Beer-Sheva, Israel
romhi@post.bgu.ac.il

Ziv Aharoni
Duke University
Durham, United States
ziv.aharoni@duke.edu

Henry D. Pfister
Duke University
Durham, United States
henry.pfister@duke.edu

Haim H. Permuter
Ben-Gurion University
Beer-Sheva, Israel
haimp@bgu.ac.il

Abstract

In this work, we adapt and analyze Neural Polar Decoders (NPDs) for end-to-end communication systems. While prior work demonstrated the effectiveness of NPDs on synthetic channels, this study extends the NPD to real-world communication systems. First, the NPD is adapted to complete OFDM and single-carrier communication systems. Then, to satisfy practical system requirements, the NPD is extended to support any code length via rate matching, higher-order modulations, and robustness across diverse channel conditions. Since the NPD operates directly on channels with memory, it exploits their structure to achieve higher data rates without requiring pilots and a cyclic prefix. Although the NPD entails higher computational complexity than the standard 5G polar decoder, its neural network architecture enables an efficient representation of channel statistics, resulting in manageable complexity suitable for practical systems. Simulation results over 5G channels demonstrate that the NPD consistently outperforms the 5G polar decoder in terms of BER, BLER, and throughput. These improvements are particularly significant for low-rate and short-block configurations, which are prevalent in 5G control channels. Furthermore, NPDs applied to single-carrier systems offer performance comparable to OFDM with lower PAPR, enabling effective single-carrier transmission over 5G channels. These results position the NPD as a high-performance, pilotless, and robust decoding solution.

1 Introduction

Polar codes, introduced by Arıkan in 2009 [5], are the first class of codes proven to achieve the capacity of binary-input symmetric memoryless channels under low-complexity decoding. In particular, the proof relies on successive cancellation (SC) decoding. Their adoption for control channels in 5G highlights their practical relevance in modern wireless systems.

In wireless communication, memory effects arise from intersymbol interference (ISI) due to multipath propagation and time variations from user mobility. To handle these effects, 5G systems rely on orthogonal frequency-division multiplexing (OFDM), interleaving, and equalization, which aim to transform the channel into a memoryless channel. These methods, however, introduce additional complexity such as channel estimation and equalization, as well as overhead from pilot symbols and the cyclic prefix (CP). Consequently, 5G systems achieve reliability at the expense of spectral efficiency and data rate.

In contrast, this work aims to tackle the problem by applying coding techniques directly over channels with memory, specifically for 5G polar code scenarios. We build on Neural Polar Decoders (NPDs),

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: AI and ML for Next-Generation Wireless Communications and Networking (AI4NextG).

which were originally developed in [4, 3] for synthetic channels with relatively short memory compared with the realistic channel models used in 5G. NPDs retain the recursive structure of successive cancellation (SC) decoding but replace the core operations with neural networks (NNs). Their complexity, $O(dhN \log N)$, is independent of channel memory, where d and h denote the input and hidden dimensions, and they can be trained directly from input–output observations without prior channel knowledge. These properties make NPDs promising candidates for improving spectral efficiency and data rate in 5G polar code scenarios. However, prior works did not evaluate NPDs in realistic 5G settings, nor did they address essential system features such as high-order modulation, rate matching, or robustness across diverse channel conditions and SNRs.

In this work, we address these gaps by extending the NPD to fully support modern communication system requirements. The proposed NPD handles high-order modulations, rate-matching, and operates robustly across diverse channel conditions. We integrate the NPD into end-to-end OFDM and single-carrier systems and evaluate it over standardized 5G TDL channels. Results show that the NPD consistently outperforms the 5G polar decoder in BER, BLER, and throughput, even without pilots and CP. Moreover, we show that NPDs enable reliable single-carrier communication over 5G channels, where single-carrier waveforms provide advantages over OFDM, including lower peak-to-average power ratio (PAPR) and reduced hardware complexity. These gains come at the cost of higher complexity and scenario-specific design, but remain practical for systems with sufficient computational resources and strict reliability demands.

Our contributions are:

- We show how to adapt NPD for end-to-end communication systems.
- We develop rate-matching for NPD, enabling compatibility with any code lengths.
- We extend NPD to support high-order modulation schemes.
- We adapt the NPD architecture, training procedure, and design to achieve robustness.

2 Notations and Preliminaries

Throughout this paper, random variables (RVs) are denoted by capital letters, and their realizations are denoted by lower-case letters, for example, X and x. Calligraphic letters denote sets, for example, $\mathcal{X}$. We use the notation X_i^j to denote the RV $(X_i, X_{i+1}, \ldots, X_j)$ and x_i^j to denote its realization. If $i = 1$, we may omit the index i to simplify the notation, that is, X^j. The probability $\Pr[X = x]$ is denoted as $P_X(x)$. The Mutual information (MI) between two RVs X, Y is denoted by $I(X;Y)$. $\mathcal{G}_{\mathsf{NN}}^{(d_i,h,d_o)}$ is class of neural networks with input d_i and output d_0 dimensions and $h \in \mathbb{N}$ neurons.

Polar code: Let U^N be formed by assigning information bits to indices in $\mathcal{A}$ and frozen bits to indices in $\mathcal{F}$. The codeword is $X^N = U^N G_N$, Arıkan's polar transform G_N [5] with block length $N = 2^n$, and Y^N is the channel output. A polar code can be described via a channel embedding E and the core components of the successive cancellation (SC) decoder, F, G, H. The term $E : \mathcal{Y} \to \mathcal{E}$ denotes the channel embedding, where $\mathcal{E} \subset \mathbb{R}^d$. The functions $F : \mathcal{E} \times \mathcal{E} \to \mathcal{E},\ G : \mathcal{E} \times \mathcal{E} \times \mathcal{X} \to \mathcal{E}$ denote the check-node and bit-node operations, respectively. We denote by $H : \mathcal{E} \to \mathbb{R}$ a mapping of the embedding into an LLR value, that is, a soft-decision. For a memoryless channel W, the channel embedding is the LLR $E(y) = \log \frac{W(y|1)}{W(y|0)}$, assuming uniform inputs. SC decoding then applies the recursive functions $F(e_1, e_2) = -2\tanh^{-1}(\tanh \frac{e_1}{2} \tanh \frac{e_2}{2})$, $G(e_1, e_2, u) = e_2 + (-1)^u e_1$, and $H(e) = e$, with hard decision $h(l) = \mathbb{I}_{l>0}$. Given channel outputs y_1^N, the decoder computes the LLR of synthetic channels

$$L_{U_i|U_1^{i-1},Y_1^N} = \log \frac{P_{U_i|U_1^{i-1},Y_1^N}(1|u_1^{i-1}, y_1^N)}{P_{U_i|U_1^{i-1},Y_1^N}(0|u_1^{i-1}, y_1^N)} \tag{1}$$

for $i \in \mathcal{A}$ and estimates $\hat{u}_i$ accordingly. For further details, see [5, Sec. VIII]

Neural Polar Decoders: The NPD introduced in [4] performs SC decoding by replacing the functions E, F, G, H with neural networks. Specifically, the embedding network $E_{\theta_E} : \mathcal{Y} \to \mathbb{R}^d$ maps the channel outputs into a d-dimensional embedding space, where $E_{\theta_E} \in \mathcal{G}_{\mathsf{NN}}^{(1,h,d)}$ for $\mathcal{Y} \subset \mathbb{R}$ and h denotes the number of hidden neurons. The SC NPD components are defined as follows: the check-node network $F_{\theta_F} \in \mathcal{G}_{\mathsf{NN}}^{(2d,h,d)}$, the bit-node network $G_{\theta_G} \in \mathcal{G}_{\mathsf{NN}}^{(2d+1,h,d)}$, and the embedding-to-LLR

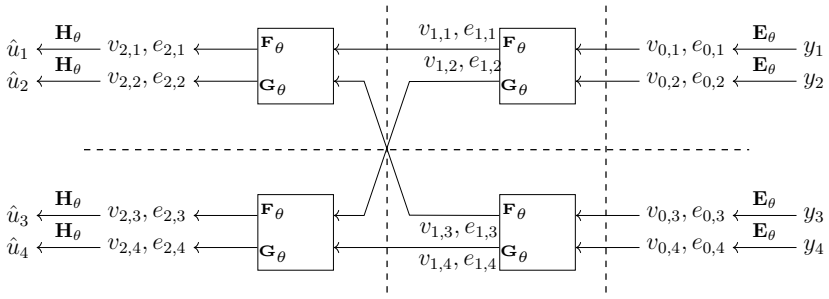


Figure 1: Block diagram of the NPD with code length $N = 4$.

network $H_{\theta_H} \in \mathcal{G}_{\mathsf{NN}}^{(d,h,1)}$. Because the NPD preserves the structure of the SC decoder, it can be naturally extended to successive cancellation list (SCL) and CRC-aided SCL (CA-SCL) decoding, as shown in [3, 12]. The NPD with list-size L has a computational complexity of $O(LhdN \log N)$. Figure 1 illustrates the NPD for code length of $N = 4$, where $e_{l,i}$ denotes the embedding of the i-th bit at decoding depth l, and $v_{l,i}$ is the corresponding ground-truth bit. E.g. $v_{n,1}^N = u^N$ and $v_{0,1}^N = x^N$.

3 Neural Polar Decoders for communication

This section extends the original NPD [4, 3] to practical communication systems. Enhancements include support for higher-order modulation, robustness across diverse SNRs, and integration of rate matching via puncturing. In addition, we adapt the decoder to both single-carrier and OFDM systems and revise the training and design procedures for practical deployment.

Channel Embedding for Communication: We extend the original NPD channel embedding to support practical systems. For higher-order modulations, the embedding network outputs m embeddings (each of dimension d) per received symbol, where m is the number of bits per symbol. To improve robustness, the estimated noise variance N_0 is included as an input, enabling adaptation across SNRs without retraining. Formally, the embedding network is a function $E_{\theta_E} : \mathbb{R}^2 \times \mathbb{R}^+ \to \mathbb{R}^{d \cdot m}$, that maps the real and imaginary parts of the received symbol and N_0 to m embeddings of dimension d.

Rate Matching for NPD: Polar codes were originally defined for lengths equal to a power of 2, whereas practical systems require support for arbitrary code lengths. To address this, we adapt puncturing strategies for NPDs. For binary phase-shift keying (BPSK), we adopt the puncturing scheme of [7], removing P bits according to the set $\mathcal{P} = (B_N)_1^P$, where the B_N is the bit-reversal permutation matrix. For higher-order modulations, entire symbols are punctured based on the bit-reversal order. For example, given $N = 8$, $P = 4$, and $m = 2$ bits per symbol, the bit-reversal order is $B_N = (1, 5, 3, 7, 2, 6, 4, 8)$. Thus, bits 1 and 5 determine the symbols to puncture, leading us to puncture the symbols containing bits $\{1, 2\}$ and $\{4, 5\}$. Consequently, the puncturing set becomes $\mathcal{P} = \{1, 2, 4, 5\}$, indicating the bits that are not transmitted. This heuristic approach for higher-order modulation achieved consistently good performance in our experiments, with potential enhancements to be investigated in future work.

At the decoder, punctured symbols are represented by a learned *constant embedding*, $E_\theta^{\text{co}} \in \mathbb{R}^{d \cdot m}$, reshaped into m embeddings of dimension d. Each constant embedding corresponds to an LLR of zero, consistent with classical puncturing. The training ensures that $H_\theta(E_\theta^{\text{co}}(\mathbf{0})) = 0$, for uniform input bits. Further details are given in the training phase.

NPD-Based Communication Systems: We present NPD-based communication systems for OFDM and single-carrier transmission. As shown in Fig. 2, both share the same encoder, rate matching, Mapper, and NPD structure, differing only in the modulation/pulse-shaping block.

The input sequence b^K is encoded by a polar encoder using the following steps. First, u^N is formed by inserting information bits in the information set and frozen bits in the frozen set, then the encoded codeword is computed as $x^N = u^N G_N$ using the polar transform. After rate matching, the codeword x^{N_r} is mapped to modulation symbols $s^{N_r/m}$ and transmitted via OFDM or single-carrier waveforms. At the receiver, demodulation yields the received sequence $y^{N_r/m}$. Decoding proceeds identically for both systems. The embedding function E_{θ^*} maps $y^{N_r/m}$ to embedding vectors $e_{0,1}^{N_r/m}$, rate recovery

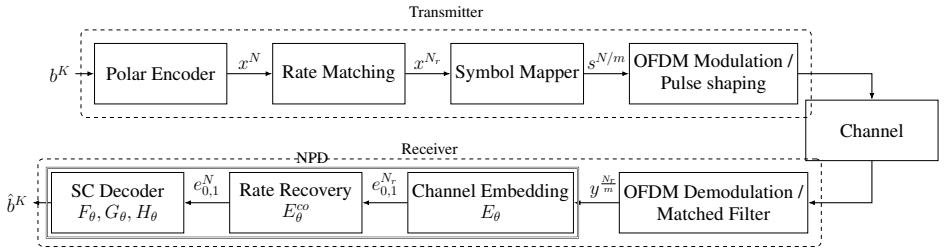


Figure 2: Block diagram of an end-to-end communication system integrating the proposed NPD.

reconstructs punctured positions using constant embeddings $E^{\text{co}}_{\theta^*}$, and NN SC decoding is performed with $F_{\theta^*}, G_{\theta^*}, H_{\theta^*}$, yielding the estimate $\hat{b}^K$.

Training Phase: The NPD parameters are trained through an iterative procedure based on stochastic gradient descent (SGD). In each iteration, a batch of channel input–output pairs is sampled and the gradient of the loss function is used to update the parameter set θ. The objective is to learn θ so that the NN SC outputs approximate the true LLRs $L_{U_i|U^{i-1},Y^N}\left(y^N, u^{i-1}\right),\ i \in [1:N]$ of the synthesized channels, while also learning an embedding representation of the punctured bit positions for the proposed rate-matching strategy.

As summarized in Algorithm 1, a block $(x^N, y^{N/m})$ is sampled and mapped to embeddings e_0 via E_θ using y and estimated noise variance N_0, while e_0^{co} is obtained from a constant channel embedding. Two losses are then computed:$\mathcal{L}_X = \text{NPDLoss}(e_0, x^N; \theta)$ and $\mathcal{L}_Y = \text{NPDLoss}(e_0^{\text{co}}, x^N; \theta)$. The loss $\mathcal{L}_Y$ aligns the decoder output with the LLRs of the synthesized channels, whereas $\mathcal{L}_X$ optimizes the embedding representation of the punctured bit. The total loss $\mathcal{L} = \mathcal{L}_X + \mathcal{L}_Y$ is minimized using SGD. The NPDLoss, introduced in [4], can be computed in $\log N$ steps [3]. It is defined as the normalized sum of cross-entropy terms:

$$\text{NPDLoss}(e_0, x^N; \theta) = \frac{1}{N(n+1)} \sum_{j=0}^{n} \sum_{i=1}^{N} L^{\theta}_{ce}(e_{j,i}, v_{j,i}), \tag{2}$$

where $e_{l,i}$ represents the embedding of the i-th bit at the l-th decoding depth, and $v_{l,i}$ is the corresponding ground truth bit. The cross-entropy loss is given by $L^{\theta}_{ce}(e, v) = -v \log H_\theta(e) - (1 - v) \log\big(1 - H_\theta(e)\big)$. Further detailed in Appendix 6.3.

Code Design: The design phase determines the information set $\mathcal{A}$ and the frozen set $\mathcal{F}$ by estimating the mutual information (MI) of the synthesized channels, $\widehat{\mathsf{I}}_{\theta^*}(U_i; Y^N|U^{i-1})$, using the optimized NPD parameters θ^*. The procedure, summarized in Algorithm 2, follows the approach in [3] and relies on a modified version of the NPDLoss 2 algorithm, denoted $\widetilde{\text{NPDLoss}}$. Rather than computing the loss, this algorithm outputs both the bits $v_n = u^N$ and their corresponding embeddings e_n, which

Algorithm 1 NPD for communication training

input: Dataset $\mathcal{D}_B$, #of iterations $\mathsf{N}_{\mathsf{iters}}$, learning rate γ
output: Optimized θ^*

Initiate the weights of $E_\theta, E^{co}_\theta, F_\theta, G_\theta, H_\theta$
for $\mathsf{N}_{\mathsf{iters}}$ iterations **do**
  Sample $x^N, y^N \sim \mathcal{D}_B$
  Compute $u^N = x^N G_N$
  Compute $e_0 \leftarrow E_\theta(y^{N/m}, N_0)$
  Duplicate $E^{co}_\theta(\mathbf{0})$ to e^{co}_0
  Compute $\mathcal{L}_X = \text{NPDLoss}(e^{co}_0, x^N; \theta)$
  Compute $\mathcal{L}_Y = \text{NPDLoss}(e_0, x^N; \theta)$
  Update $\theta := \theta - \gamma \nabla_\theta(\mathcal{L}_X + \mathcal{L}_Y)$
end for
return θ^*

Algorithm 2 Code Design for NPD

input: θ^*, k number of information bits
output: $\mathcal{A}, \mathcal{F}$
  Generate $x^N \sim \text{Bern}(0.5)$
  Apply rate matching to obtain x^{N_r}
  Receive $y^{N_r/m}$
  Compute $e^{N_r}_{0,1}$ using $E_\theta(y^{N_r/m}, N_0)$
  if $N \neq N_R$ **then**
    Apply rate recovery to obtain $e^N_{0,1}$
  end if
  Compute $[\mathbf{v}_n, \mathbf{e}_n] = \widetilde{\text{NPDloss}}\left(e^N_{0,1}, x^N; \theta^*\right)$
  Compute $\widehat{\mathsf{I}}_{\theta^*}\left(U_i; Y^N|U^{i-1}\right)$
  Compute $Q^N_1 = \mathsf{Argsort}\left(\widehat{\mathsf{I}}_{\theta^*}\right)$
  return $\mathcal{A} = Q^{k-1}_1, \mathcal{F} = Q^N_k$

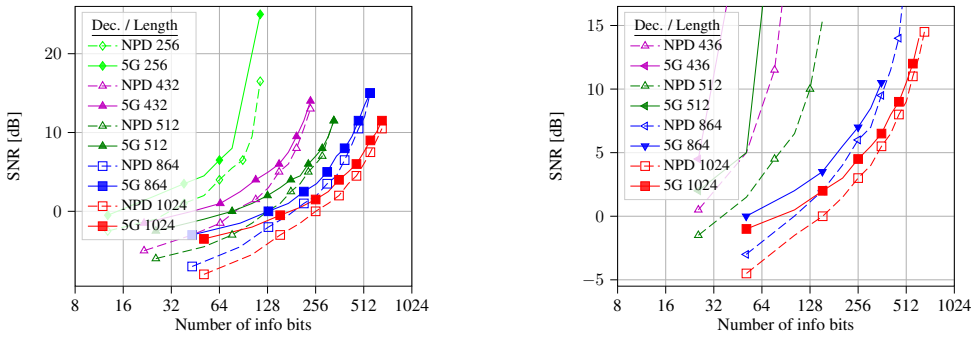
are then used to estimate the MI as

$$\widehat{I}_{\theta^*}(U_i; Y^N | U^{i-1}) = \frac{1}{B} \sum_{(x^N, y^{N_r/m}) \in \mathcal{D}_B} \Big(1 - L^{\theta}_{ce}(e_{n,i}, v_{n,i})\Big). \tag{3}$$

4 Experiments and insights

The experiments were conducted on single-input single-output (SISO) systems using both OFDM and single-carrier transmissions. The communication channel based on the discrete-time baseband model for the wireless channel presented in [13], using the 3GPP TDL profiles [11]. Our implementation was developed in Python using TensorFlow [2] for NN modeling and the Sionna library for communication system simulations [8]. Full setup details about the training and evaluation are in the Appendix. 6.1.

Performance of NPD vs. 5G Polar in OFDM Systems: We compare the proposed NPD with the 5G polar decoder over OFDM, considering both power-of-two and rate-matched non-power-of-two code lengths, under BPSK and QPSK modulations. Simulations were conducted on a TDL-A channel with 100 ns delay spread and user mobility of 0–8 m/s. The 5G decoder follows 3GPP rate matching, while the NPD use the proposed NPD rate matching scheme.



(a) BPSK (b) QPSK

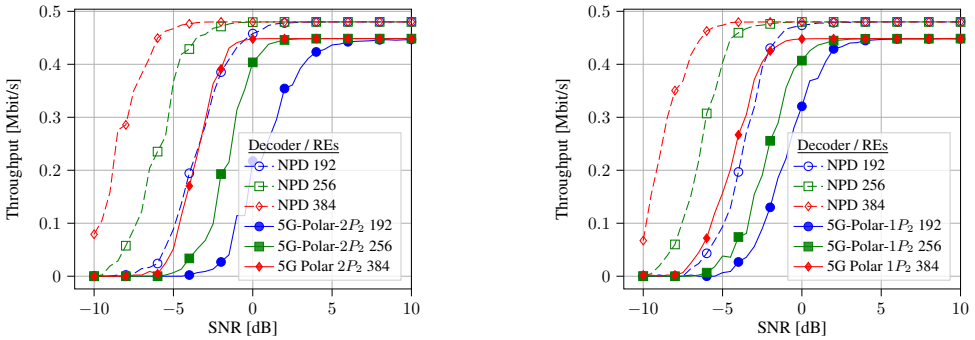
Figure 3: SNR required to achieve BLER $= 10^{-3}$ under OFDM transmission over a TDL-A channel.

From Figures 3b and 3a, we can see that the NPD consistently outperforms the 5G polar decoder across all code lengths and modulations (BPSK and QPSK). Notably, the NPD achieves this gain without using pilot symbols and a CP, thereby reducing the transmission overhead, thus increasing spectral efficiency. The performance advantage becomes increasingly pronounced as the code rate, which operates in the low SNR regime. One possible reason for this gain is that the 5G Polar uses least-squares (LS) channel estimation from the pilots, whereas the NPD can learn to extract better channel estimates. Moreover, at low SNR, where channel memory dominates, the NPD achieves additional gains by effectively mitigating these memory effects.

Throughput Comparison: This experiment compares the throughput of the NPD and the 5G polar decoder under identical resource element (RE) allocations per frame to ensure a fair comparison. Each frame consists of eight OFDM symbols, with the number of subcarriers determined by the total RE budget. For instance, with 256 REs per frame, the system uses 32 subcarriers. In the 5G polar decoder, the $1P_2$ and $2P_2$ configurations (Fig.6) introduce pilot overheads of 16 and 32 REs, respectively. Throughput is computed as: Throughput $= (1 - \text{BLER}) \cdot \frac{k}{T}$, where $k = 32$ is the number of information bits, and $T = \frac{1}{\text{subcarrier spacing}} + T_{\text{cp}}$ denotes the total frame duration. For the 5G decoder, the CP duration is $T_{\text{cp}} = 4.69\,\mu$s, whereas the NPD operates without a CP ($T_{cp} = 0$).

These experiments, shown in Figure 4, present the throughput versus SNR results for both high- and low-Doppler conditions across various REs per frame, and demonstrate that the NPD significantly outperforms the 5G polar decoder in terms of throughput. This gain is attributed to the NPD's superior BLER performance, its ability to employ longer code lengths (since the 5G decoder reserves REs for pilots), and its operation without a CP, which provides an additional throughput improvement of approximately 7% compared to systems using a normal CP.

Specifically, Figure 4a (high-Doppler, user velocities 15–30m/s) shows that the 5G polar decoder employs the 2P configuration to maintain robustness under rapid channel variation, whereas Figure4b



(a) High Doppler

(b) Low Doppler

Figure 4: Throughput vs. SNR for transmitted information bits $k = 32$ using different REs per frame.

(low-Doppler, velocities 0–8 m/s) shows the use of the 1P configuration to improve spectral efficiency. In both scenarios, the NPD consistently delivers higher throughput across all SNR levels and RE configurations.

Comparison of Single-Carrier and OFDM Systems: This experiment compares the performance of the NPD and 5G polar decoder in single-carrier and OFDM-based systems. The systems use BPSK modulation and a code length of $N = 1024$, with user mobility in the range of 0–8 m/s. For the single-carrier waveform, a sinc filter was used for pulse shaping. This uses the same bandwidth as the corresponding OFDM configuration to ensure a fair comparison.

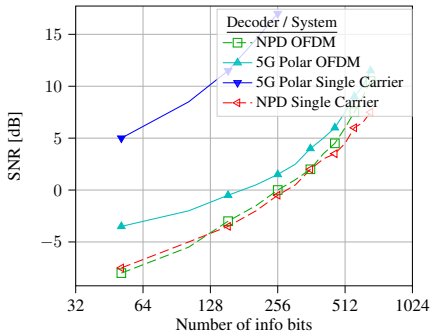


Figure 5: Required SNR to achieve a target BLER of 10^{-3} for single-carrier and OFDM systems.

Figure 5 shows that the proposed NPD enables single-carrier systems to achieve decoding performance comparable to that of OFDM, even under severe multipath conditions and without increasing complexity. This is particularly impactful, given the inherent advantages of single-carrier waveforms, such as lower PAPR and simpler hardware requirements. The reduced PAPR improves the power amplifier efficiency under nonlinear conditions. These results highlight the potential of NPD-based single-carrier systems as viable alternatives to OFDM.

Limitations: Despite the performance gains demonstrated in this work, several limitations should be noted. First, the computational complexity of NPDs is higher than that of the classical 5G polar decoder, though it remains practical on modern hardware. Second, the proposed approach requires scenario-specific code design for each modulation and SNR regime, whereas 5G polar codes rely on a universal reliability sequence defined by the standard. Finally, NPDs require an initial training phase, which introduces additional computational overhead. However, once trained, inference can be performed without pilots.

5 Conclusions

In this study, we presented extended NPDs tailored for practical communication systems. The proposed NPD operates directly over memory channels without relying on pilots and CP. NPDs offer a robust and reliable solution with a manageable decoding complexity. The NPD was enhanced to support higher-order modulation, rate matching for any code lengths, and robust performance

across various channel conditions. Evaluations over 5G channels show that the NPD consistently outperforms the 5G polar decoder in terms of BER, BLER, and throughput. In particular, it shows significant gains in the low rate and short code length that are typical of control channels. Moreover, single-carrier systems with NPD achieve performance comparable to OFDM while benefiting from lower PAPR and improved amplifier efficiency. Overall, the NPD offers a pilotless, high-performance decoding solution with a strong potential for future wireless systems.

References

[1] 3rd Generation Partnership Project (3GPP). *Multiplexing and Channel Coding*. Technical Specification 3GPP TS 38.212 version 15.3.0. 3GPP, 2018.

[2] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems*. 2016. arXiv: 1603.04467 [cs.DC]. URL: https://arxiv.org/abs/1603.04467.

[3] Ziv Aharoni et al. "Code Rate Optimization via Neural Polar Decoders". In: *2024 IEEE International Symposium on Information Theory (ISIT)*. 2024, pp. 2424–2429. DOI: 10.1109/ISIT57864.2024.10619429.

[4] Ziv Aharoni et al. "Data-Driven Neural Polar Codes for Unknown Channels With and Without Memory". In: *arXiv preprint arXiv:2309.03148* (2023).

[5] E. Arikan. "Channel Polarization: A Method for Constructing Capacity-achieving Codes for Symmetric Binary-input Memoryless Channels". In: *IEEE Trans. Inf. Theory* 55.7 (2009), pp. 3051–3073.

[6] Valerio Bioglio, Filippo Gabry, and Ingmar Land. "Design of polar codes in 5G new radio". In: *IEEE Communications Surveys & Tutorials* 23.1 (2020), pp. 29–40.

[7] Valerio Bioglio, Frederic Gabry, and Ingmar Land. "Low-Complexity Puncturing and Shortening of Polar Codes". In: *2017 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. IEEE, 2017, pp. 1–6. DOI: 10.1109/WCNCW.2017.7919115.

[8] Jakob Hoydis et al. *Sionna: An Open-Source Library for Next-Generation Physical Layer Research*. arXiv preprint arXiv. Mar. 2022.

[9] Kai Niu and Kai Chen. "CRC-aided decoding of polar codes". In: *IEEE Communications Letters* 16.10 (2012), pp. 1668–1671. DOI: 10.1109/LCOMM.2012.081512.120662.

[10] Tom Richardson and Ruediger Urbanke. *Modern Coding Theory*. Cambridge: Cambridge University Press, 2008. ISBN: 9780521852296.

[11] *Study on channel model for frequencies from 0.5 to 100 GHz*. Technical Report TR 38.901. Version 16.1.0. Release 16. 3rd Generation Partnership Project (3GPP), 2020. URL: https://www.3gpp.org/ftp/Specs/archive/38_series/38.901/38901-g10.zip.

[12] I. Tal and A. Vardy. "List Decoding of Polar Codes". In: *IEEE Trans. Inf. Theory* 61.5 (2015), pp. 2213–2226.

[13] D. Tse and P. Viswanath. *Fundamentals of Wireless Communication*. Cambridge, UK: Cambridge University Press, 2005.

6 APPENDIX

This appendix documents the experimental setup and implementation details used in the paper. Section 6.1 specifies the training and evaluation configurations and hyperparameters (Table 1), the OFDM frame structure and pilot patterns (Fig. 6), and the 5G polar system blocks (Fig. 7). Section 6.3 provides the full definition and step-by-step computation of NPDLoss used in both the training and code-design phases.

6.1 Experiment Setup

This section describe the experiment configuration. Table 1 lists the parameters used for training and evaluation.

Model Evaluation: Throughout the experiments, we compared the proposed NPD against the standard 5G uplink polar code [1] that relies on conventional equalization and channel estimation, both employing CRC-assisted list decoding (CA-SCL). Training was conducted using TDL-C power

Table 1: Parameters Used for Training and Evaluation

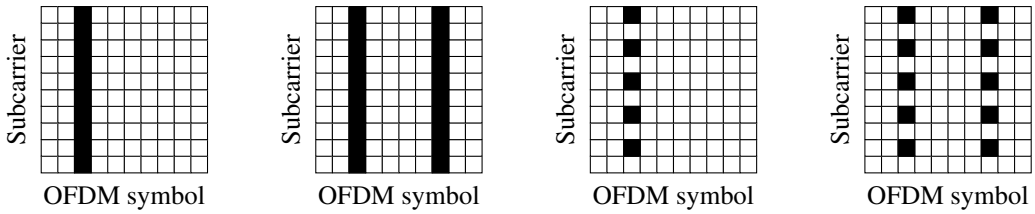
Parameter	Value
Training and Model Configuration	
Power delay profile	TDL-C (300 ns)
Noise variance range (uniform)	–5 to 15 dB
Velocity range (uniform)	0 to 35 m/s
Batch size	100
Training iterations	10^6
code length	1024
Embedding dimension (d)	128
Neurons units (h)	300
Learning rate	10^{-3}
Optimization algorithm	Adam
OFDM Configuration	
Carrier frequency	3.5 GHz
Subcarrier spacing	15 kHz
Classic System Configuration	
OFDM channel estimation	Least Squares
OFDM equalizer	Linear MMSE
single-carrier channel estimation	Perfect
single-carrier equalizer	Zero Forcing
5G Polar configuration	Uplink
Cyclic Prefix	$4.69\mu s$
Evaluation Configuration	
Power delay profile	TDL-A
List decoder size	16
CRC length	11

delay profiles, and evaluation was performed on TDL-A profiles to avoid overfitting to a specific channel model.

A single model per modulation scheme and system waveform (OFDM or SC) was trained and used across all channel conditions and code sizes in the evaluation, demonstrating the generalization capability of the model without requiring retraining or fine-tuning.

Frame Structure: In OFDM experiments with BPSK and a code size of 1024, the system uses eight OFDM symbols and 128 subcarriers. For smaller code sizes (864, 512, 432, 256, and 216), the number of subcarriers was set to 108, 64, 54, 32, and 27, respectively. In the QPSK experiments, 64 subcarriers were used for a code size of 1024 and 32 for 512. In the 5G polar code OFDM-based system, the CP is configured to maintain an approximate 7% overhead. For instance, with a code size of 1024 and 128 subcarriers, a CP length of 9 yields $T_{\text{cp}} = 4.68\mu$ s, closely matching the 3GPP normal CP of 4.69μ s at 15kHz spacing.

The 5G polar decoder was evaluated under four pilot configurations, as shown in Figure 6. In the "$1P$" and "$2P$" settings, the pilots occupy all subcarriers in one or two OFDM symbols, respectively. In the "$1P_2$" and "$2P_2$" settings, pilots are placed on every second subcarrier within the designated symbols. Unless stated otherwise, the $2P$ configuration was used by default. All pilots were randomly generated using QPSK modulation. Note that when comparing 5G polar and NPD at the same code length, extra OFDM symbols were added for pilots in the 5G decoder based on the chosen configuration.



(a) Pilot pattern "$1P$" (b) Pilot pattern "$2P$" (c) Pilot pattern "$1P_2$" (d) Pilot pattern "$2P_2$"

Figure 6: Pilot patterns used in simulations.

Design: The information set $\mathcal{A}$ for is found using the design phase of NPD for each SNR point, modulation, and code length. It assumes knowledge of the design condition at the encoder for

selecting the design. In contrast, the 5G decoder relies on a static reliability sequence as specified in the 3GPP standard [1]. This provides a deployment advantage for the standard 5G decoder.

Training Configuration: This training configuration was carefully designed to balance performance and robustness while maintaining a manageable computational complexity. A thorough exploration of the training hyperparameters was conducted to determine an effective setup for the proposed NPD. This investigation included variations in the embedding dimension, number of neurons, batch size and learning rate.

To develop a robust model, training was performed under challenging conditions, including a delay spread of 300ns, which resulted in a model capable of generalizing across a wide range of delay spreads. To ensure resilience to varying SNRs, the training dataset was generated using noise variances sampled uniformly from the range $[-5, 10]$ dB. Furthermore, to improve robustness to Doppler effects, the training channels included user velocities uniformly distributed between 0 and 35m/s.

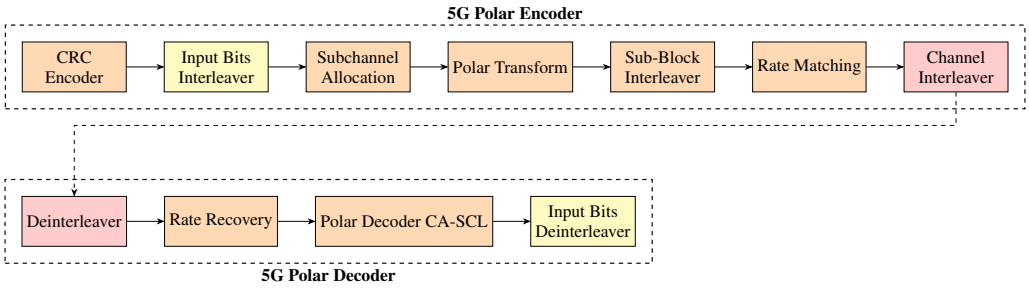


Figure 7: 5G Polar code system. Yellow, red, and orange blocks are implemented in downlink, uplink, and both, respectively.

6.2 5G Polar Code

The 5G polar encoder begins with a cyclic redundancy check (CRC) encoder that appends a 24-bit (downlink) or 11-bit (uplink) CRC to the input bits. In the downlink, an input-bit interleaver is applied before subchannel allocation, which constructs the polar input vector u^N by placing the information and CRC bits in the information set $\mathcal{A}$ and zeros in the frozen set $\mathcal{F}$. The polar transform then generates the codeword $x^N = u^N F^{\otimes n}$. Notably, the bit-reversal permutation B_N is not included. The codeword is then passed through a subblock interleaver and a rate-matching unit, which adjusts the length to the desired number of transmission bits $N_r \leq N$ via puncturing, shortening, or repetition [10]. In the uplink, the bits are further permuted by a channel interleaver, whereas in the downlink, this step is omitted.

In the 5G polar decoder, processing begins with a deinterleaver in the uplink, whereas this step is omitted in the downlink. This is followed by a rate recovery stage that reconstructs the original codeword length N, resulting in a length-N LLR vector. These LLRs are then decoded using CRC-aided successive cancellation list (CA-SCL) decoding [9]. In the downlink, an additional input bit deinterleaver is applied after decoding to restore the original bit order. For further details, refer to [1, 6].

6.3 NPDLoss

This subsection presents the algorithm used to compute the NPDLoss in $\mathcal{O}(\log N)$ steps. The method is based on the algorithm in [3], with the modification that the input is the embedding e_0 rather than y^N. Consequently, the embedding function is applied outside the loss computation.

Given x^N and the embeddings e_0, the loss of the NPD is computed as follows. the log likelihood ratios (LLRs) are computed as $\mathbf{l}_0 = \mathbf{H}_\theta(\mathbf{e}_0)$. The loss of bits at stage 0 is then computed as

$$\mathcal{L}_0(\mathbf{v}_0, \mathbf{e}_0; \theta) = \\ -\frac{1}{N}\sum_{i=1}^{N} v_{0,i} \log \sigma(l_{0,i}) + \overline{v}_{0,i} \log \overline{\sigma(l_{0,i})}, \tag{4}$$

Algorithm 3 NPDLoss

Input:
e_0, x^N
$F_\theta, G_\theta, H_\theta$ ▷ NPD model
Output:
$\mathcal{L}(x^N, y^N; \theta)$ ▷ computed loss

Stage 0:
$\mathbf{v}_0 \leftarrow x^N$
$\mathbf{l}_0 \leftarrow \mathbf{H}_\theta(\mathbf{e}_0)$

$$\mathcal{L}_0(\mathbf{v}_0, \mathbf{e}_0; \theta) = -\frac{1}{N}\sum_{i=1}^{N} v_{0,i} \log \sigma(l_{0,i}) + \overline{v}_{0,i} \log \overline{\sigma(l_{0,i})}$$

▷ loss of stage 0

Stage 1:
$\mathbf{v}_1 \leftarrow [\mathbf{v}_0^o \oplus \mathbf{v}_0^e \mid \mathbf{v}_0^e]$
$\mathbf{e}_1 \leftarrow [\mathbf{F}_\theta(\mathbf{e}_0^o, \mathbf{e}_0^e) \mid \mathbf{G}_\theta(\mathbf{e}_0^o, \mathbf{e}_0^e, \mathbf{v}_0^o \oplus \mathbf{v}_0^e)]$
$\mathbf{l}_1 \leftarrow \mathbf{H}_\theta(\mathbf{e}_1)$

$$\mathcal{L}_1(\mathbf{v}_1, \mathbf{e}_1; \theta) = -\frac{1}{N}\sum_{i=1}^{N} v_{1,i} \log \sigma(l_{1,i}) + \overline{v}_{1,i} \log \overline{\sigma(l_{1,i})}$$

▷ loss of stage 1

Stages 2 to n:
for $j = 2$ to n **do**
Compute $\mathbf{v}^{(j-1)}$ and $\mathbf{e}^{(j-1)}$ ▷ Equation (8)
Initiate $\mathbf{v}^{(j)} = \emptyset, \mathbf{e}^{(j)} = \emptyset, \mathbf{l}^{(j)} = \emptyset$
for each $\tilde{\mathbf{v}}_{j-1} \in \mathbf{v}^{(j-1)}$ and $\tilde{\mathbf{e}}_{j-1} \in \mathbf{e}^{(j-1)}$ **do**
Compute $\tilde{\mathbf{v}}_j$, $\tilde{\mathbf{e}}_j$ and $\tilde{\mathbf{l}}_j$ ▷ Equation (5)–(7)
$\mathbf{v}^{(j)} \leftarrow \mathbf{v}^{(j)} \cup \tilde{\mathbf{v}}_j, \mathbf{e}^{(j)} \leftarrow \mathbf{e}^{(j)} \cup \tilde{\mathbf{e}}_j, \mathbf{l}^{(j)} \leftarrow \mathbf{l}^{(j)} \cup \tilde{\mathbf{l}}_j$
end for
Concatenate $\mathbf{v}^{(j)}, \mathbf{e}^{(j)}, \mathbf{l}^{(j)}$ into $\mathbf{e}_j$, $\mathbf{v}_j$, $\mathbf{l}_j$

$$\mathcal{L}_j(\mathbf{v}_j, \mathbf{e}_j; \theta) = -\frac{1}{N}\sum_{i=1}^{N} v_{j,i} \log \sigma(l_{j,i}) + \overline{v}_{j,i} \log \overline{\sigma(l_{j,i})}$$

▷ loss of stage j

end for
return $\mathcal{L}(e_0, x^N; \theta) \leftarrow \frac{1}{n+1}\sum_{j=0}^{n} \mathcal{L}_j(\mathbf{v}_j, \mathbf{e}_j; \theta)$

where $\overline{x} = 1 - x$, $\sigma(x) = \frac{1}{1+e^{-x}}$ is the logistic function and $P_{U_i|U^{i-1},Y^N}\left(1|u^{i-1}, y^N\right) = \sigma\left(L_{U_i|U^{i-1},Y^N}\left(u^{i-1}, y^N\right)\right)$.

At stage 1, the loss is computed in the following manner. First, $\mathbf{v}_1$ is computed by

$$\mathbf{v}_1 = [\mathbf{v}_0^o \oplus \mathbf{v}_0^e \mid \mathbf{v}_0^e], \tag{5}$$

where $\mathbf{v}_0^o, \mathbf{v}_0^e \in \mathbb{F}_2^{\frac{N}{2}}$ contain the odd and even elements of $\mathbf{v}_0$, respectively. Next, $\mathbf{e}_1$ is computed as follows:

$$\mathbf{e}_1 = [\mathbf{F}_\theta\left(\mathbf{e}_0^o, \mathbf{e}_0^e\right) \mid \mathbf{G}_\theta\left(\mathbf{e}_0^o, \mathbf{e}_0^e, \mathbf{v}_0^o \oplus \mathbf{v}_0^e\right)], \tag{6}$$

where $\mathbf{F}_\theta\left(\mathbf{e}_0^o, \mathbf{e}_0^e\right), \mathbf{G}_\theta\left(\mathbf{e}_0^o, \mathbf{e}_0^e, \mathbf{v}_0^o \oplus \mathbf{v}_0^e\right) \in \mathbb{R}^{\frac{N}{2} \times d}$, the operator $[\cdot|\cdot]$ denotes the concatenation of two matrices along the first dimension, and $\mathbf{e}_1 \in \mathbb{R}^{N \times d}$. The loss of stage 1 is then computed by first computing

$$\mathbf{l}_1 = \mathbf{H}_\theta\left(\mathbf{e}_1\right), \tag{7}$$

and then computing $\mathcal{L}_1\left(\mathbf{v}_1, \mathbf{e}_1; \theta\right)$, as done in stage 0.

At stages $j \in [2:n]$, the same computations as in stage 1 are followed, but they are performed within sub-blocks independently. Given $\mathbf{e}_{j-1}$ and $\mathbf{v}_{j-1}$, we first split them into collections:

$$\begin{aligned} \mathbf{v}^{(j-1)} &= \{\mathbf{v}_{j-1,k}\}_{k=0}^{2^{j-1}-1} \\ \mathbf{e}^{(j-1)} &= \{\mathbf{e}_{j-1,k}\}_{k=0}^{2^{j-1}-1}, \end{aligned} \tag{8}$$

where

$$\mathbf{v}_{j-1,k} = \{v_{j-1,l}\}_{l=1+k2^{n-j+1}}^{(k+1)2^{n-j+1}}$$
$$\mathbf{e}_{j-1,k} = \{e_{j-1,l}\}_{l=1+k2^{n-j+1}}^{(k+1)2^{n-j+1}},$$

with $\mathbf{v}_{j-1,k} \in \mathbb{F}_2^{\frac{N}{2^{j-1}} \times 1}$ and $\mathbf{e}_{j-1,k} \in \mathbb{R}^{\frac{N}{2^{j-1}} \times d}$. For every $\mathbf{v}_{j-1,k}$ and $\mathbf{e}_{j-1,k}$, we repeat the computations in Equations (5)–(6) to produce $\mathbf{v}'_{j-1,k}$ and $\mathbf{e}'_{j-1,k}$, which are then concatenated into $\mathbf{e}_j$, $\mathbf{v}_j$. Next, $\mathbf{e}_j$, $\mathbf{v}_j$ are used to compute $\mathcal{L}_j\left(\mathbf{v}_j, \mathbf{e}_j; \theta\right)$ and are passed to the next stage. The overall loss is computed by

$$\mathcal{L}\left(x^n, y^n; \theta\right) = \frac{1}{n+1} \sum_{j=0}^{n} \mathcal{L}_j\left(\mathbf{v}_j, \mathbf{e}_j; \theta\right), \tag{9}$$

and the corresponding gradient is given by $\nabla_\theta \mathcal{L}\left(x^n, y^n; \theta\right)$. The loss computation is summarized in Algorithm 3.