The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering and Mathematics (EPSTEM), 2025

Volume 37, Pages 574-589

ICEAT 2025: International Conference on Engineering and Advanced Technology

Learning About Syndrome Awareness and WMS Algorithm for Adaptive Neural Decoding for 6G LDPC Base Graph Enhancement

Noor Salih Mohammed
Al-Furat Al-Al-Awsat Technical University

Ahmed Ghanim Wadday
Al-Furat Al-Al-Awsat Technical University

Bashar Jabbar Hamza
Al-Furat Al-Al-Awsat Technical University

Abstract: 6G wireless communication networks require low-latency, ultra-reliable error correction capabilities that adapt to dynamic channel scenarios and block lengths. This study presents a new adaptive neural decoding workflow for 5G-New Radio LDPC base graphs (BG1 and BG2). It combines a weighted Min-Sum (WMS) algorithm with syndrome-aware learning that uses parity-check feedback to guide neural decoding in two key ways: syndrome-based loss and syndrome-conditioned message updates. This approach overcomes the limitations of traditional min-sum (MS) and belief propagation (BP) decoders. To improve message updates among iterations based on real-time parity-check feedback, we first introduce trainable parameters for better training on the most challenging error patterns; a unique data pipeline collects "uncorrected" frame samples at low SNR and generates log-likelihood ratios (LLRs) under the Additive White Gaussian Noise (AWGN) channel and Rayleigh flat-fading channel. The training scheme has two stages: (1) an end-to-end supervised stage focused on minimizing both soft-BER and soft-FER loss across random noisy codewords, and (2) a boosting stage for learning residual corrections using mean-squared error on uncorrected frames. Performance is evaluated across various SNR levels, lifting factors, and code rates. Results show that BG1 outperforms BG2 in the AWGN channel by 1 dB and in the Rayleigh flat-fading channel by 2 dB. To balance reliability and decoding complexity, early convergence is achieved within 10 to 20 iterations. Additionally, lower rates and higher lifting factors produce sharper waterfalls and error floors below 10^{-7} and 10^{-8} for AWGN and Rayleigh, respectively. The proposed framework generalizes to different channel types and LDPC designs and offers a 0.3 dB waterfall gain compared to traditional neural Min–Sum decoders. These results demonstrate adaptable, high-performance error correction suitable for various wireless applications, highlighting the practicality of syndrome-aware WMS neural decoding for future 6G standards.

Keywords: Base Graph-LDPC codes optimization, 6G-Neural LDPC decoder, Weighted min-sum decoding, Syndrome-aware learning, Uncorrected-frame boosting

Introduction

Advanced communication networks, such as 5G New Radio (NR) commonly utilize low-density parity-check (LDPC) codes as the coding scheme for data channels due to their excellent reliability and effectiveness (Richardson & Kudekar, 2018; TSGR, 2020) in achieving performance close to the Shannon limit (Chung et al., 2001). LDPC codes employ iterative decoding (Fossorier, 2001) with multiple algorithms available. Sum standard sum-product (SP) algorithm is also known as the belief-propagation (BP) algorithm. Due to its computational complexity (Sun & Jiang, 2018), the BP technique achieves nearly optimal decoding performance. However, the min-sum (MS) decoding algorithm (Kschischang et al, 2002), despite being slightly less efficient, is a suitable choice due to its significantly lower complexity.

The transition toward 6G introduces additional issues, including 1. Dynamic Channel Conditions: Enhanced mobility and various applications require adaptable and resilient error correction. 2. Shorter Block Lengths: In ultra-low-latency applications, LDPC codes must perform efficiently with shortened codewords. 3. Scalability: The LDPC codes should effectively adapt to diverse block lengths, coding rates, and applications (Rowshan et al., 2024). To overcome this issue, various algorithms have been employed, including the offset min-sum and normalized min-sum (Chen et al., 2005; Wu et al., 2010), to optimize the MS decoding algorithm. The goal is to achieve near-perfect performance with the BP algorithm while minimizing complexity. Additionally, they are typically optimized for fixed channels like Additive White Gaussian Noise (AWGN), which reduces robustness in fading scenarios, such as Rayleigh fading. To ensure reliable communication in real-world applications, it is necessary to develop intelligent and adaptive decoding algorithms that can generalize across various uncertain channel conditions. Traditional MS decoding algorithms, due to their computational performance, exhibit considerable sub-optimality because they utilize uniform and constant weights across all decoding iterations. This inflexible structure is unsuitable for dynamic channel scenarios, causing degraded decoding effectiveness, particularly among non-Gaussian noise or channel distortions (Nachmani et al., 2017).

Recent developments in neural decoding algorithms have focused on combining learnable parameters during the decoding procedure. However, they tend not to leverage key syndrome information. This data indicates the direct mathematical feedback given by the LDPC parity-check equations. Therefore, conventional neural decoders do not automatically modify their message-passing updates across decoding iterations and edges in response to real-time signs of the decoding process's success or failure. In Dai et al. (2021) the researchers developed a neural Min-Sum (NMS) decoder strategy that combines iteration-by-iteration training and parameter sharing to address vanishing gradient problems and lower complexity during training.. The authors in Kwak et al. (2024) introduced boosted NMS decoders using a novel training methodology that utilizes two stages of the neural decoding process, where the second decoder is trained specifically to correct errors when the initial decoder fails to correct. A hybrid framework that uses standard neural MS variants combined with ordered statistics decoding OSD, developed by the authors in (Li & Yu, n.d.) through adaptive procedures, including dynamic error pattern grouping and iteration diversity, aims to enhance decoding beyond the error floor. Due to the lack of systematic training procedures for NMS decoding, as well as performance and training efficiency issues, the authors in Na et al. (2025) suggested optimal training methodologies integrated with dataset creation. These methodologies improve decoding performance beyond traditional techniques, particularly in the error floor region, while ensuring minimal computational complexity.

The primary goal of the proposed system is to optimize decoding performance by developing an adaptive neural decoder structure based on a weighted Min-Sum (WMS) framework. The optimization and contributions of this study are summarized as follows:

1. Develop a decoding framework for NMS with trainable parameters to improve message updating between the variable and check nodes, enabling adaptive message passing based on communication conditions.
2. Integrate syndrome-aware learning to boost decoder performance and early convergence by enabling it to switch weight updates according to real-time parity-check feedback dynamically.
3. The custom database workflow generates log-likelihood ratio (LLR) samples and simulates LDPC code performance across multiple channel models, collecting uncorrected real-time samples to highlight the learning framework in challenging decoding scenarios.

The proposed framework aims to enhance training efficiency, significantly improve decoding performance, and increase model robustness compared to random uniform sampling, while generalizing across multiple channel types and providing a scalable solution for future wireless standards, such as 6G and beyond. This paper compares the performance of base graph (BG) LDPC code by evaluating BG1 and BG2, code lengths and code rates impacts on BG2 across AWGN, and Rayleigh fading channels, as well as the transmitting modulation technique of quadratic phase shift keying (QPSK).

LDPC Base Graph Construction

The $[N, K]$ LDPC code involves N bits for the code length, K bits for the information length, and $M = N - K$ parity bits. The sparse parity-check matrix H is commonly used to represent the LDPC code, which has dimensions $M \times N$ and a code rate of $R = K/N$. The codeword c is represented by a binary vector of length N. The set of all codewords indicates the code. The code satisfies the condition $c \cdot H^T = 0$. As demonstrated in the Tanner

(1981) graph presents a complete illustration of LDPC codes that helps the understanding of decoding methods. The Tanner graph is a bipartite graph representation that defines two sets of nodes: N variable nodes (VN) of a set $\{v_0, v_1, \dots, v_{N-1}\}$, and M check nodes ($CN$) of a set $\{c_0, c_1, \dots, c_{M-1}\}$. An edge exists between the corresponding $VN\ v_j$ and $CN\ \ c_i$ only if $H_{i,j} = 1$. The 5G-NR system utilizes two distinct base matrices, BG1 and BG2 (Petrović et al., 2021). Without these blocks, we cannot construct the LDPC code or verify its error-correcting capabilities. The standards (Kumar et al., 2023) specify fixed sizes for both base graphs: BG1 measures 46×68, and BG2 measures 42×52. Table 1 details the criteria for BG1 and BG2 according to the 3GPP TS 38.212 standard (Nguyen et al., 2019).

Table 1. The 5G-NR LDPC BG parameters

Parameter	BG 1	BG 2
Base Graph dimension	46×68	42×52
Code rate	$1/3 \le R \le 8/9$	$1/5 \le R \le 2/3$
Systematic bits columns	22	10
No. of non-zero elements	316	197
Information lengths	$500 \le K \le 8448$	$40 \le K \le 2560$
The sub-Matrices sizes	**A**: 4×22 ; **E**: 4×4 ; **O**: 4×42 **B**: 42×22 ; **C**: 42×4 ; **I**: 42×42	**A**: 4×10 ; **E**: 4×4 ; **O**: 4×38 **B**: 38×10 ; **C**: 38×4 ; **I**: 38×38

The matrix fixed block structure is divided into columns consisting of three parts: information columns, core parity columns, and extension parity columns. Rows are subdivided into core check and extension check, as shown in Fig. 1. The matrix structure comprises submatrices A and B, which together form the kernel (the information part), while the submatrices E, O, C, and I are referred to as (extensions) for parity bit calculation.

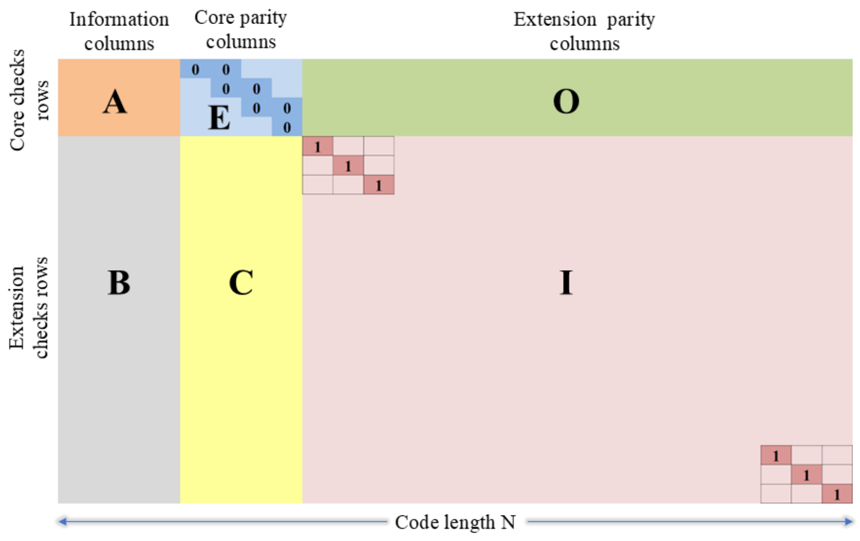


Figure 1. Base graph structure

LDPC codes consist of multiple expansion factors, all of which are determined by the index i_{LS}, which organizes the expansions. Equation (1) establishes the highest value of 384 for the expansion factor Z_C. Each element in the base graph features a circularly shifted value ranging between -1 and 383. Two variables determine the expansion, factor a and factor J_a, as described by the equation (1) and Table 2 for Z_C corresponding to 3GPP TS38.212 standard (3GPP, 2020).

$$Z_C = a \times 2^j \qquad (1)$$

Where $\{a : 2,3,5,7,9,11,13,15\}$ and the value of $\{j: 0,1,2, \cdots, J_a\}$, based on the set index ($i_{LS}: 0, 1, 2, 3, 4, 5, 6, 7$). The H matrix is a crucial component of LDPC codes, used for both encoding and decoding. To construct H, we replace every entry of the BG by a square matrix $Z_C \times Z_C$. Generating the H matrix for LDPC codes is a meticulous process that must strike a balance between the requirements for sparsity, error-correcting capability, and computational efficiency. The process of transforming BG into an H matrix suitable for LDPC codes requires three main steps as follows:

• Step 1: Replace (-1) entries in BG with a zero matrix of size $Z_C \times Z_C$.

• Step 2: Replace (0) entries in BG with an identity matrix I_Z.

Table 2. The LDPC- Z_C sets

Set index (i_{LS})	Set of Lifting Sizes (Z_C)
0	$Z_C = 2 \times 2^j,\ j = 0,1,2,3,4,5,6,7 \to \{2, 4, 8, 16, 32, 64, 128, 256\}$
1	$Z_C = 3 \times 2^j,\ j = 0,1,2,3,4,5,6,7 \to \{3, 6, 12, 24, 48, 96, 192, 384\}$
2	$Z_C = 5 \times 2^j,\ j = 0,1,2,3,4,5,6 \to \{5, 10, 20, 40, 80, 160, 320\}$
3	$Z_C = 7 \times 2^j,\ j = 0,1,2,3,4,5 \to \{7, 14, 28, 56, 112, 224\}$
4	$Z_C = 9 \times 2^j,\ j = 0,1,2,3,4,5 \to \{9, 18, 36, 72, 144, 288\}$
5	$Z_C = 11 \times 2^j,\ j = 0,1,2,3,4,5 \to \{11, 22, 44, 88, 176, 352\}$
6	.$Z_C = 13 \times 2^j,\ j = 0,1,2,3,4 \to \{13, 26, 52, 104, 208\}$
7	$Z_C = 15 \times 2^j,\ j = 0,1,2,3,4 \to \{15, 30, 60, 120, 240\}$

• Step 3: Replace (i) entries in BG with an identity matrix I_Z,but with a right shift performed i times, where i ranges from 0 to $Z_C - 1$.

The suggested parameters for BG1 and BG2 in this research are $i_{LS} = 0$ and $Z_C = 8$; meaning that the range of entries extends from -1 to 7. Every element in the BGs is converted into an 8×8 identity matrix that forms H. Table 3 outlines the specifications and corresponding values of BG 1 and BG 2, while Fig. 2 presents the configuration of the structured BG1.

Table 3.Parameters and related values

Parameter	BG 1	BG 2
Block length (n_b)	68	52
Information bits count (k_b)	22	10
Block rows	46	42
Edges	316	197
Columns weight (w_v)	1 to 30	37 to 41
Rows weight (w_c)	3 to 19	47 to 51
Code rates	0.32	0.19
H field	$\mathbb{F}_2^{368\times544}$	$\mathbb{F}_2^{336\times416}$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	……	67	68
1	73	15	103	49	-1	240	39	-1	-1	15	162	215	164	133	-1	298	110	-1	……	-1	-1
2	303	-1	294	27	261	161	-1	133	4	80	-1	129	300	-1	76	266	72	83	……	-1	-1
3	68	7	80	-1	280	38	227	202	200	71	106	-1	-1	295	283	301	-1	184	……	-1	-1
4	220	208	-1	30	197	-1	61	175	79	-1	281	303	253	164	53	-1	44	28	……	-1	-1
5	233	205	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	……	-1	-1
6	83	292	-1	50	-1	-1	-1	-1	-1	-1	-1	-1	318	-1	-1	-1	201	-1	……	-1	-1
7	289	-1	-1	-1	-1	-1	21	-1	-1	-1	293	13	-1	232	-1	-1	-1	302	……	-1	-1
8	12	88	-1	-1	207	-1	-1	50	25	-1	-1	-1	-1	-1	76	-1	-1	-1	……	-1	-1
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	……	:	:
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	……	:	:
45	234	-1	-1	-1	-1	-1	-1	227	-1	259	-1	-1	-1	-1	-1	-1	-1	-1	……	0	-1
46	-1	101	-1	-1	-1	-1	228	-1	-1	-1	126	-1	-1	-1	-1	-1	-1	-1	……	-1	0

Figure 2. Base graph 1 matrix

Neural Min-Sum Decoding for Base Graph LDPC Codes

The min-sum (MS) decoding algorithm serves as an iterative decoding algorithm in which data are exchanged between variable nodes and check nodes with each iteration, having the message update criteria for check nodes close to the basic minimum operation (Na et al., 2025). To boost the algorithm's performance, the weighted MS (WMS) decoder technique (Kwak et al., 2024) was proposed, which incorporates an α (normalization factor) into the CN messages. The NMS decoding technique (Dai et al., 2021) intends to improve the decoding performance by utilizing α as a learnable parameter and incorporating various weighting factors for iteration t. The WMS is an iterative decoding procedure that passes data in the form of log-likelihood ratios (LLRs) via the edges of the $Tanner\ graph$. The LLR of the v_{th} bit in a transmitted binary codeword bit $x \in \{0,1\}$ is determined by comparing the noisy received signal vector y to x, given by:

$$L_v = log \frac{Pr(y_v|x_v = 0)}{Pr(y_v|x_v = 1)} \quad (2)$$

$log(\cdot)$ stands for "logarithm base 2", and $Pr(receive\ y_v|sent\ x_v)$ is the channel's transition probability (or density) from the transmitted bits x to the received vector y illustrates the whole behavior of the channel, including bit flips, noise, and fading. At iteration t in the iterative decoding process, the message that passes from $(VN\ v \rightarrow CN\ c)$ is:

$$L_{v\rightarrow c}^{(t)} = L_v + \sum_{c' \in \mathcal{N}(v)/c} \hat{L}_{c'\rightarrow v}^{(t-1)} \quad (3)$$

Where $\mathcal{N}(v)$ is the sets of neighboring nodes $VN\ v$, c' indicates each of the other CNs linked to the same $VN\ v$, unless this $CN\ c$ is where message updates are computed. The message passes $(CN\ c \rightarrow VN\ v)$, $\hat{L}_{c\rightarrow v}^{(t)}$:

$$\hat{L}_{c\rightarrow v}^{(t)} = 2\tanh^{-1}\left(\prod_{v' \in \mathcal{N}(c)/v} \tanh\left(\frac{L_{v'\rightarrow c}^{(t)}}{2}\right)\right) \quad (4)$$

$\mathcal{N}(c)$ is neighboring node set of $CN\ c$, and v' indicates all of the other VNs linked to the same $CN\ v$,except $VN\ v$. The initial values are $L_{v\rightarrow c}^{(0)} = L_{ch}$,and $\hat{L}_{c\rightarrow v}^{(0)}$=0, where L_{ch} is the channel LLR. At first iteration ($\hat{L}_{c'\rightarrow v}^{(t-1)} = 0$) in (3), ($L_{v\rightarrow c}^{(0)} = L_v) = L_{ch}$. After t iterations of message-passing decoding, the soft estimation s_v "best guess" of a posteriori LLR for x bits in (2) being zero, compared to one, as:

$$s_v = L_v + \sum_{c' \in \mathcal{N}(v)} \hat{L}_{c'\rightarrow v}^{(t)} \quad (5)$$

The eq. (5) represents the combination of initial channel information L_v and extrinsic information from all neighboring parity checks $\hat{L}_{c'\rightarrow v}^{(t)}$.

The decision rule provided in (6) that converts the soft LLR estimate s_v into a binary bit estimate $\hat{x}_v$,as:

$$\hat{x}_v = \frac{1 - sgn(s_v)}{2} \quad (6)$$

Here's how the sign function of soft estimation s_v $(sgn(s_v))$ operates:

$$sgn(s_v) = \begin{cases} +1, & s_v > 0 \\ -1, & s_v < 0 \\ 0, & s_v = 0 \end{cases}$$

The iterative decoding methodology explained previously is known as the belief-propagation (BP) algorithm or the sum-product (SP) algorithm. To decrease computing complexity in equation (4), caused by hyperbolic tangent functions and numerous multiplications, the MS approach is employed to derive equation (4) as:

$$\hat{L}_{c\rightarrow v}^{(t)} = \left(\prod_{v' \in \mathcal{N}(c)/v} sgn\left(L_{v'\rightarrow c}^{(t)}\right)\right) \times \min_{v' \in \mathcal{N}(c)/v} \left|L_{v'\rightarrow c}^{(t)}\right| \quad (7)$$

In contrast to the SP approach, the MS approximation incurs a non-negligible efficiency loss. Other advanced MS algorithms were released in (Dai et al., 2021). Thus, in this study, the neural MN (NMS) (Kwak et al., 2024) is adopted, incorporating the channel weight $w_{ch}^{(t)}$, into equation (3) and the check node weight $w_{c\rightarrow v}^{(t)}(\alpha)$ for iteration t into equation (7) . So, the $(VN\ v \rightarrow CN\ c)$ update sublayer is written as:

$$L_{v\to c}^{(t)} = w_{ch}^{(t)} L_{ch} + \sum_{c' \in \mathcal{N}(v)/c} \hat{L}_{c'\to v}^{(t-1)} \tag{8}$$

And $(CN\ c\ \to VN\ v)$ update sublayer as:

$$\hat{L}_{c\to v}^{(t)} = \alpha^{(t)} \times \left(\prod_{v' \in \mathcal{N}(c)/v} sgn\left(L_{v'\to c}^{(t)}\right) \right) \times \min_{v' \in \mathcal{N}(c)/v} \left| L_{v'\to c}^{(t)} \right| \tag{9}$$

Finally, the v_{th} element of the output LLR vector L_v^o is calculated at the last iteration T given by:

$$L_v^o = L_v^{ch} + \sum_{c' \in \mathcal{N}(v)} \hat{L}_{c'\to v}^{(t)} \tag{10}$$

The primary objective of this framework is to optimize the weights of the NMS decoder to improve the performance of the BG-LDPC code.

Proposed BG-LDPC Code Scheme

This section outlines the proposed system model design methodology step-by-step, as illustrated in Fig. 3. It also describes the neural network structure for training weights of the NMS decoder.

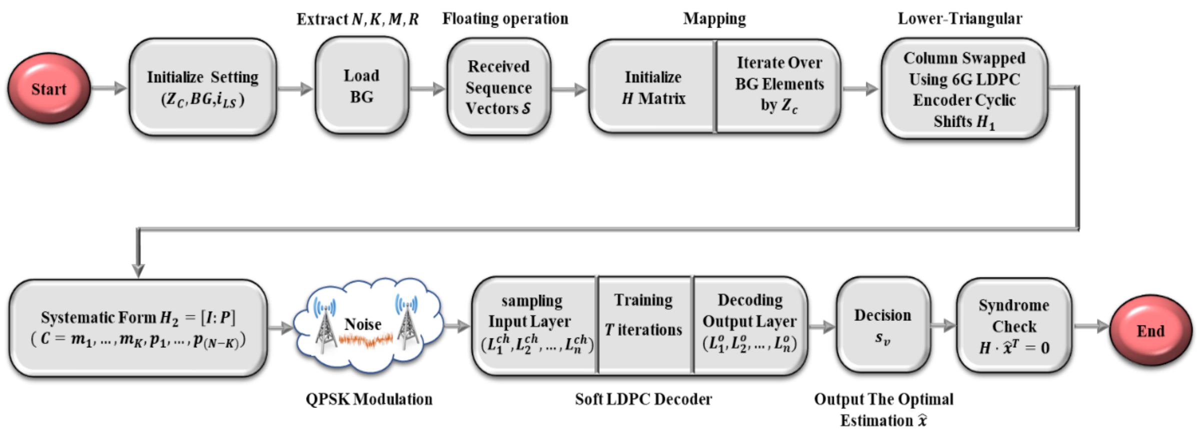
The System Model

The overall structure of the proposed adaptive WNMS decoder's end-to-end process description is illustrated by the flowchart in Fig. 3. The initial step is collecting datasets. This study trains its neural network using a dataset of LDPC BGs and their associated BER. The details for these datasets derive from the 3GPP TS 38.212 standards. After the data collection step, the parameters Z_C , BG (BG1 or BG2), R, and i_{LS} are initialized. The BG has been loaded and converted to a flat form (sequence vectors $\mathcal{S}$) to construct the $H \in \{0,1\}^{M\times N}$ matrix after these parameters are parsed and fed into this mapping function. Circular right shifts are then applied based on 14.

Based on shift values Z_C to encode the message using a lower triangular LDPC encoder in GF (2) to obtain a systematic form of $H_2 = [I: P]$ and concatenate the message and parity bits to form the codeword($C = m_1, \dots, m_K, p_1, \dots, p_{(N-K)}$). The specified channels influence the transmitted QPSK signals (AWGN, and Rayleigh flat fading channel). The NMS algorithm is implemented through channel weights to enhance the LDPC code-based BG matrix. The decision is made by (6) is to obtain the optimal estimation $\hat{x}$ and tested by applying the syndrome rule $H \cdot \hat{x}^T = 0$ as the last step.

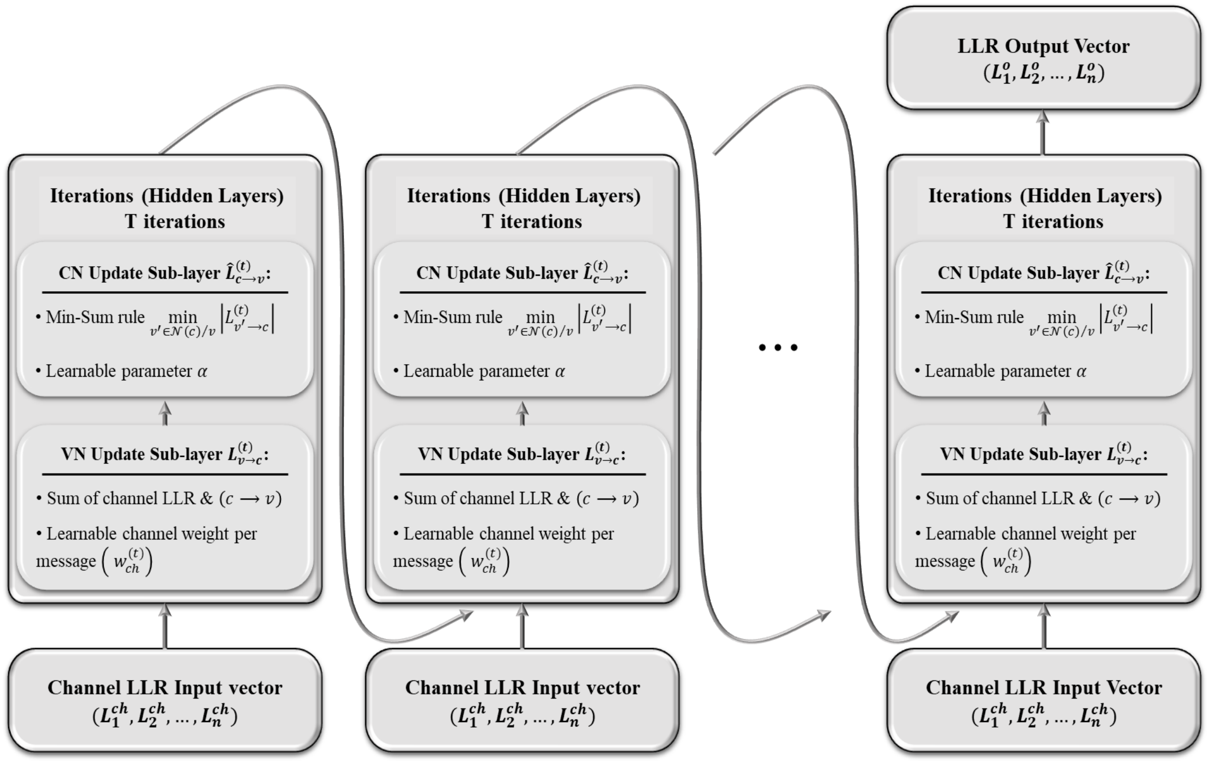
Decoder Training Methodology

The training procedure implemented in our work is based on a supervised learning technique called gradient-based optimization. Specifically, we are applying Adam optimization (Kingma & Ba, 2015) to update the learnable parameters, which include the weight and bias tensors of a neural network-based LDPC decoder. Here's a detailed description of the neural network structure illustrated in Fig. 4. The neural network's input layer is a vector of channel LLR $\{L_1^{ch}, L_2^{ch}, \dots, L_n^{ch}\}$ having the same length as the number of $VNs\ v$ in the $Tanner\ graph$, and it produces the output layer LLR vector $\{L_1^o, L_2^o, \dots, L_n^o\}$, also containing the same number of neurons. All hidden layers share the same size, which is equal to the number of edges in the Tanner graph . Two hidden layers (sublayers) are allocated for each decoding iteration t, with each containing one neuron corresponding to each edge in the Tanner graph. The variable node update $(VN\ v \to CN\ c)$ messages in (8) are produced by the odd hidden layers, while the even layers transmit the check node update $(CN\ c\ \to VN\ v)$ messages in (9). The input layer connected to all odd hidden layers represents the incorporation of the channel LLR in (8). The weight and bias tensors are updated using the Adam optimizer, a variant of gradient descent,are: the channel weight $w_{ch}^{(t)}$, the check node weight $\{\alpha\}_{t=1}^{T}$,the VN edge weights $\{w_{ij}^{vn}\}$ along with the optional biases b_{vn}, b_{cn}.



Start
Initialize Setting (Z_C, BG, i_{LS})
Extract N, K, M, R
Load BG
Floating operation
Received Sequence Vectors S
Mapping
Initialize H Matrix
Iterate Over BG Elements by Z_c
Lower-Triangular
Column Swapped Using 6G LDPC Encoder Cyclic Shifts H_1
Systematic Form $H_2 = [I: P]$ ($C = m_1, \dots, m_K, p_1, \dots, p_{(N-K)}$)
Noise
QPSK Modulation
sampling Input Layer ($L^{ch}_1, L^{ch}_2, \dots, L^{ch}_n$)
Training T iterations
Decoding Output Layer ($L^o_1, L^o_2, \dots, L^o_n$)
Soft LDPC Decoder
Decision s_v
Output The Optimal Estimation $\hat{x}$
Syndrome Check $H \cdot \hat{x}^T = 0$
End

Figure 3.Diagram of the adaptive neural decoding procedures



LLR Output Vector ($L^o_1, L^o_2, \dots, L^o_n$)
Iterations (Hidden Layers) T iterations
CN Update Sub-layer $\hat{L}^{(t)}_{c\to v}$:
• Min-Sum rule $\min_{v'\in\mathcal{N}(c)/v} \left|L^{(t)}_{v'\to c}\right|$
• Learnable parameter α
VN Update Sub-layer $L^{(t)}_{v\to c}$:
• Sum of channel LLR & ($c \to v$)
• Learnable channel weight per message ($w^{(t)}_{ch}$)
Channel LLR Input vector ($L^{ch}_1, L^{ch}_2, \dots, L^{ch}_n$)
• • •

Figure 4.The network diagram of the proposed NMS decoder

Collectively, these parameters define the forward pass:

- The $(VN\ v)$ update is:

$$L^{(t)}_{v\to c} = w^{vn}_{ij} \times \left(w^{(t)}_{ch} L^0_v + \sum_{c'\in\mathcal{N}(v)/c} \hat{L}^{(t-1)}_{c'\to v} \right) \qquad \forall i \in v, \qquad j \in \mathcal{N}(v) \qquad (11)$$

- The $(CN\ c)$ update is:

$$\hat{L}^{(t)}_{c\to v} = \alpha^{(t)} \times \left(\prod_{v'\in\mathcal{N}(c)/v} sgn\left(L^{(t)}_{v'\to c}\right) \right) \times \min_{v'\in\mathcal{N}(c)/v} \left|L^{(t)}_{v'\to c}\right| \qquad \forall t \in \{1,2,\dots,T\} \qquad (12)$$

To train all learnable parameter weights $w_{ch}^{(t)}$, w_{ij}^{vn}, and α end-to-end while minimizing a syndrome-aware loss over a dataset that is randomly generated from synthetic noisy codewords, we feed a collection of samples into the neural network. After preparing the samples, generate noisy frame outlines as follows:

1- For each SNR value γ ,draw a batch of B random information bits $x \in \{0,1\}^K$.
2- Encode via the H matrix and modulate (e.g., QPSK).
3- Pass through the channel (AWGN, and Rayleigh fading channel) to obtain received vector y.

For training, we generate a random received vector y of length N, along with its corresponding channel LLR vector $\{L_1^{ch}, L_2^{ch}, \dots, L_n^{ch}\}$. After computing the channel LLRs in (2), the posteriori LLR (after T iterations) is:

$$L_v^o = w^{ch} L_v^{ch} + \sum_{c' \in \mathcal{N}(v)} \hat{L}_{c' \to v}^{(T)} \qquad (13)$$

Consequently, we used loss functions in deep learning-based decoders, including the soft BER (Lian et al., 2019)and FER (Xiao et al., 2021) loss functions.

- The soft BER:

$$\mathcal{L}_{BER} = \frac{1}{N} \sum_{v=1}^{N} \sigma(-L_v^o) \qquad (14)$$

Where the sigmoid function $\sigma(x) = (1 + e^{-x})^{-1}$ and $e = 2.71828$.

- For FER loss function

$$\mathcal{L}_{FER} = \frac{1}{B} \sum_{b=1}^{B} \frac{1}{2} \left[1 - sgn(\min_j L_v^o) \right] \qquad (15)$$

b is the number of frames.

The behavior of the $sgn(\cdot)$ function:

- The forward propagation is:

$$sgn\left(\min_j L_v^o\right) = \frac{2}{1 + e^{-sgn\left(\min_j L_v^o\right)}} - 1 \qquad (16)$$

- The Gradient backpropagation according to the Adam optimizer is:
  - Updating weights:

$$w_{new}^{(t)} \leftarrow w_{old}^{(t)} - \alpha \frac{\partial \mathcal{L}}{\partial w^{(t)}} \qquad (17)$$

$\frac{\partial \mathcal{L}}{\partial w^{(t)}}$ refers to the gradient of the loss function $(\mathcal{L}_{BER}, \mathcal{L}_{FER})$ with respect to the weight.

  - updating biases:

$$b_{new}^{(t)} \leftarrow b_{old}^{(t)} - \alpha \frac{\partial \mathcal{L}}{\partial b^{(t)}} \qquad (18)$$

In addition, the BER could vary significantly with a varying number of iterations throughout the training process, so the SNR must be allocated to different levels during training. Additionally, it is essential to note that both the trained weights and biases are shared across all lifted versions of the same base graph and distributed across every possible lifted code generated by the same base code. Algorithm 1 summarizes the suggested training techniques for the NMS decoder.

Algorithm 1: Training The Neural LDPC Decoder: End-to-End Execution Logic

```
Inputs:
$H \in \{0,1\}^{M \times N}$: parity-check matrix
$L_{ch}^{(0)}$ $[1..N]$: input LLRs: $L_v = \log(Pr(y_v|x_v = 0)/Pr(y_v|x_v = 1))$
$T$: max number of iterations (hidden layers)
$w_{ch}$: channel scaling weight (scalar)
```

```
$w_{ij}^{vn}$: VN-edge weights for each edge $i \to j$
$\alpha[1..T]$: learned CN-scaling factors per iteration
**Output:** $\hat{x}\,[1..N]$ : estimated codeword (0/1) or declare failure
**step 1. Initialize all CN→VN messages to zero**
for each check node $j$ and each neighbor $i \in M(j)$:
  $M_{cn}\,[j \to i] \leftarrow 0$
**step 2. Iterative Message Passing**
for $t$ from 1 to $T$ do
  # 2.1 VN Update Sublayer: compute variable-to-check messages
  for each variable node $i$ and each neighbor $j \in N(i)$ do
    $sum\_in \leftarrow 0$
    for each $k \in N(i) \setminus \{j\}$ do
      $sum\_in \leftarrow sum\_in + M_{cn}[k \to i]$
    end for
    # weighted sum of channel LLR and incoming CN messages
    $M_{vn}[i \to j] \leftarrow w_{ij}^{vn} * (w_{ch} * L_{ch}^{(0)} + sum\_in)$
  end for
  # 2.2 CN Update Sublayer: compute check-to-variable messages
  for each check node $j$ and each neighbor $i \in M(j)$ do
    # product of signs
    $sign_{prod} \leftarrow 1$
    for each $m \in M(j) \setminus \{i\}$ do
      $sign_{prod} \leftarrow sign_{prod} * sign(M\_vn[m \to j])$
    end for
    # minimum magnitude
    $min_{val} \leftarrow +\infty$
    for each $m \in M(j) \setminus \{i\}$ do
      if $abs(M_{vn}[m \to j]) < min_{val}$ then
        $min\_val \leftarrow abs(M_{vn}[m \to j])$
      end if
    end for
 # $learned\ CN\text{-}scaling\ factors\ per\ iteration\ - min - sum\ update$
    $M_{cn}[j \to i] \leftarrow \alpha[t] * sign_{prod} * min_{val}$
  end for
end for
**step 3. A-Posteriori LLR Computation**
for each variable node $i$ do
  $L_{app}[i] \leftarrow w_{ch} * L_{ch}^{(0)}[i]$
  for each $j \in N(i)$ do
    $L_{app}[i] \leftarrow L_{app}[i] + M_{cn}[j \to i]$
  end for
end for
end for
Feed $y$ into the NMS decoder corresponding
to the base code $Cj$ after $i-th$ iteration for the training
values of SNR and obtain output $s_v$;
Compute loss function based on (14), (15);
Update $\alpha^{(K)}$using gradient descent algorithm;
**Step4. Decision & Syndrome Check**
for $i$ from $1$ to $N$ do
  $\hat{x}\,[i] \leftarrow (1 - sign(s_v))\,/\,2$     # maps positive $LLR \to 0$, negative $\to 1$
end for
if $H \cdot \hat{x}^{T} == 0$ then
  return $\hat{x}$   # decoding successful
else
  return $\hat{x}$   # or declare decoding failure
end if
```

Boosting Learning with Uncorrected Vectors

Traditional end-to-end training strategies may perform poorly when noisy frames reach the correctable thresholds. We improve the decoding process by reutilizing the incorrect frames. The decoding process consists of two main stages: the base stage covers iteration$\{0,1,\ldots,T_1\}$ where the $VN\ v$ update multiplies by the learnable w_{ij}^{vn}, and the post stage covers iteration $\{T_1,\ T_1+1,\ldots,\bar{T}\}$, applying MS corrections using α. The post-decoder, built on boosting-learning theory, aims to mitigate the base-decoder's faults by learning from its errors. The decoding process is theoretically divided into two stages, but in practice, it performs as one decoder using learning weights using α (base) and w_{ij}^{vn} (boosting). There are four main steps in our boosting training, as follows:

Step 1. Collect uncorrected samples: Run the pretrained NMS decoder on a large test set of noisy codewords at low SNR, and collect frames where decoding fails (non-zero syndrome). Save their intermediate a posteriori LLR vectors for each frame, which serve as the input LLRs $L_i^{(0)}$ and uncorrected vectors (base-decoder output) $L_{un}^{(T)}$.
Step 2: Secondary training set: Residual target computation, in which we define the "ideal" (noiseless) reference LLRs vector as:

$$L_i^* = \ log\frac{Pr(y_i|x_i=0)}{Pr(y_i|x_i=1)} \qquad (no\ channel\ noise) \qquad (19)$$

The residual learning corrections as:

$$\Delta L = L^* - L_{un}^{(T)} \quad \in \mathbb{R}^N \qquad (20)$$

Where $\mathbb{R}$ is d-dimensional input space of codeword N.

Step 3. Boosting network stage: Repeat the same message-passing structure (NMS framework) with the learning rate and while feeding $L_{un}^{(T)}$ and the original L^0as (boost) network f_Θ, whose goal is to predict the output $\widehat{\Delta L}$ from the uncorrected frames $L_{un}^{(T)}$ and optionally L^0.

- Boost network prediction:

$$\widehat{\Delta L} \ = f_{\Theta_{boost}}\left(L_{un}^{(T)}, L^0\right) \qquad \in \mathbb{R}^N$$

$f_\Theta(\cdot)$ represents the forward function of the NMS-LDPC decoder, where Θ encompasses all the learnable parameters: $\left\{w_{ch}\ \in \mathbb{R}, \{w_{ij}^{vn}\}_{(i,j)\epsilon\mathcal{E}}, \{\alpha^{(t)}\}_{t=1}^{T}\right\}$ .This function serves as a mathematical mapping from input to output, taking noisy observations, processing all $VN\ v$ and $CN\ c$ sublayers (with their per-edge and per-iteration weights), and producing updated beliefs.

- Boosting loss: Train $f_{\Theta_{boost}}$ by minimizing the mean squared error (MSE) over a batch of size B for each of the b uncorrected frames, as (Cestari et al., 2024):
-

$$\mathcal{L}_{boost} = \frac{1}{BN}\sum_{b=1}^{B}\left\|\Delta L^{(b)} - \widehat{\Delta L}^{\,(b)}\right\|_2^2 = \frac{1}{BN}\sum_{b=1}^{B}\sum_{i=1}^{N}\left(\Delta L^{(b)} - \widehat{\Delta L}^{\,(b)}\right)^2$$

Step 4. Corrected LLR and the decision: After training, there are two stages of inference (base decoder and boost correction) to correct each uncorrected output:

- Base decoder: $L_{un}^{(T)} = f_{\Theta_{base}}(L^0)$
- Boost correction: $\widehat{\Delta L} \ = f_{\Theta_{boost}}\left(L_{un}^{(T)}, L^0\right), \quad L_{corr}^{(T)} = L_{un}^{(T)} + \widehat{\Delta L}$

This two-stage technique effectively eliminates "stuck" errors, resulting in an additional 0.1– 0.2 dB enhancement in waterfall performance. To map that belief into the final hard decision (0/1) estimate $\hat{x}$:

- Hard decision: $\hat{\boldsymbol{x}}_i = \frac{1-sgn(L_{corr}^{(T)})}{2}$

This concludes the mathematical explanation of the boost-learning process by Syndrome Check:

$$H \cdot \hat{x}^{\mathrm{T}} = 0 \qquad\qquad (decoded\ successfully)$$

Simulation Results

Proposed Training Scheme

For the BG-LDPC codes BG1 and BG2, with $i_{LS} = 0$ and $Z_c = 8$, we have selected decoding for evaluation purposes using the adaptive WMS-NMS decoder in an end-to-end training process. We employ a supervised, gradient-based approach that involves the message update rules in (11) and (12), which jointly optimize all learnable parameters (channel scaling α, $w_{ch}^{(t)}$, and w_{ij}^{vn}). The weights update method is iteration-by-iteration. The general workflow is shown in Algorithm 1 and Fig. 4.

Performance Comparison

We evaluate the proposed adaptive neural decoder (base + base-post) under AWGN and Rayleigh channel scenarios, BG configurations, lifting factors, and maximum number of decoding iterations. We collected 30000 vectors and 10000 for training, with 5000 for testing and 5000 for validation. The maximum number of iterations was also set to 20 for all codes. At each SNR point, we collect at least 100 frame errors to measure the FER performance. The scenario of transmitting coded QPSK symbols under varying channel conditions is assumed.

BG1 vs. BG2: AWGN channel

The BER and FER for BG1 and BG2 over an AWGN channel were compared using different values of SNR (2 - 5) dB, as illustrated in Fig.5. In the waterfall region, BG1 is better than BG2 by about 1 dB, with BG1 achieving a BER of approximately 1.33×10^{-7} at 4 dB, while BG2 requires 5 dB to reach a comparable BER of approximately 1.26×10^{-7}. Additionally, regarding the error floor, both graphs indicate negligible error floors down to BER $\approx\ 10^{-7}$ and FER $\approx\ 10^{-6}$, demonstrating the effectiveness of the two-stage neural boosting decoder in pushing error rates into the ultra-reliable scheme($< 10^{-7}$).

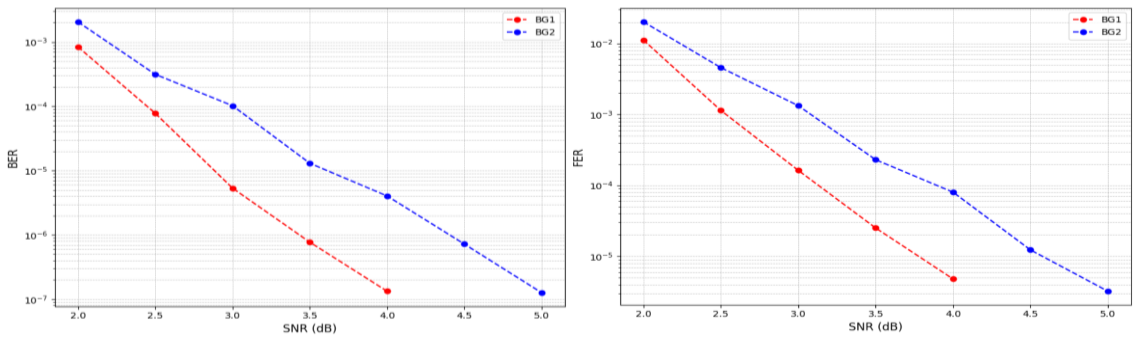


Figure 5.Performance of BG-LDPC over AWGN channel for BER and FER

BG1 vs. BG2: Rayleigh Flat-Fading Channel

Under Rayleigh fading, Fig. 6 shows both codes start with especially high error rates at 2 dB, indicating significant deep-fade effects. However, BG1's structure utilizes diversity more effectively when the SNR exceeds approximately 3 dB, leading to a much stronger waterfall effect. BG1 consistently achieves lower BER and FER across all SNRs, demonstrating more reliable frame decoding. BG1 outperforms BG2 with an approximate 2 dB improvement in BER. Additionally, BG1 enhances frame reliability compared to BG2. Below 5 dB, BG1 maintains noteworthy performance in its error floor, which decreases significantly for both BER and FER, indicating that BG1 offers better resistance to strong fades and a sharper descent approach into the ultra-reliable system under Rayleigh fading.

Investigating the Impact of Code Rate

We examine the effect of code rate on code performance, as illustrated in Fig. 7, which shows the (a) BER and (b) FER curves of BG2 for three LDPC codes ($k = 10, Z_C = 8$) with rates R = 0.19, 0.5, and 0.75 for the AWGN channel. The LDPC codes and their rates over the Rayleigh fading channel are shown in Fig. 8.

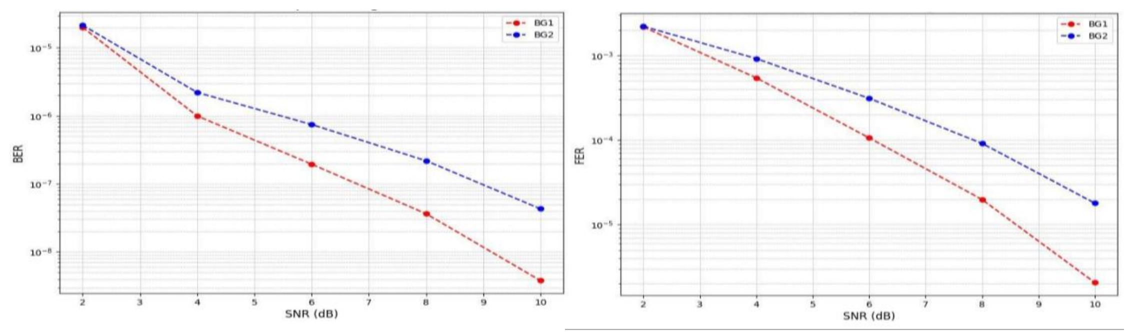


Figure 6. Performance of BG-LDPC over Rayleigh flat-fading channel for BER and FER

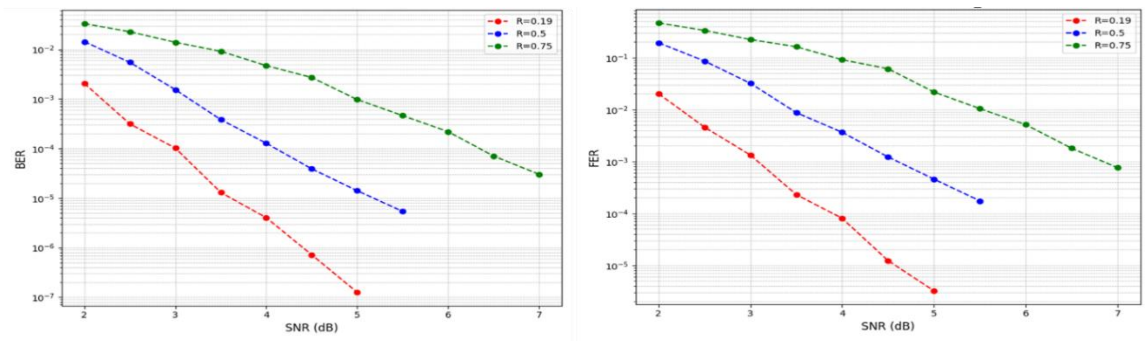


Figure 7. Comparison of code rates for AWGN channel for BER and FER

Error correction is improved by more redundancy (lower R), meaning the error rate curve shifts to the right for higher SNR. For the highest rate value (BG2, AWGN at R=0.75), we achieve BER = 3.03e-05 and FER = 7.58e-04 at 7 dB, while (BG2, Rayleigh) requires SNR = 10 to reach BER = 4.87e-04 and FER = 2.20e-02. The R=0.19 code (416, 80) outperforms R=0.5 (160, 80) by up to 1.1dB and R=0.75 (104, 80) by up to 3.34 dB in the waterfall at BER= 10^{-4} for AWGN channel.

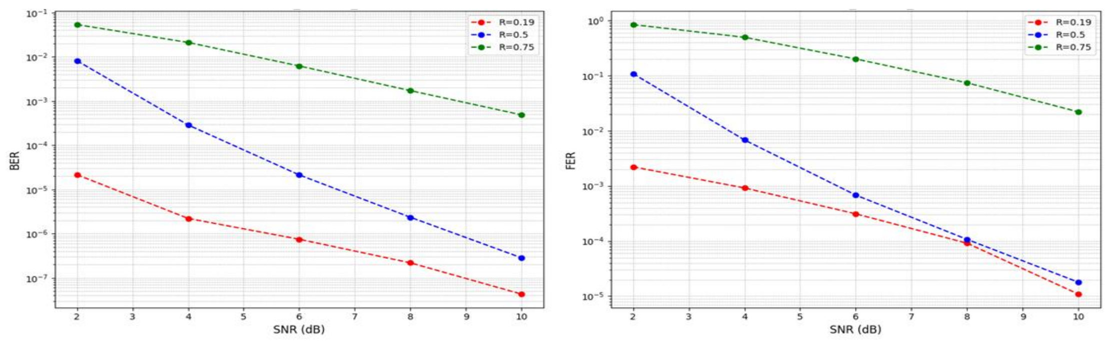


Figure 8. Comparison of code rates for Rayleigh fading channel for BER and FER

Block Length Effect

We are increasing the block length N by varying the lifting factor Z_C according to Table 2 for a fixed BG2 ($R = 0.19, i_{LS} = 0, k = 10$) in Fig. 9. The values were chosen at index $i_{LS} = 0$ ($Z_C = 2,4,8,16,32$), which produce ($N = 104,208,416,832,1664$), respectively. As Z_C grows, performance improves significantly as N increases and cycle distributions become better. For BG2, BER falls from 3.94×10^{-4} ($Z_C = 2$) to 1.50×10^{-8} ($Z_C = 32$), while FER decreases from 2.08×10^{-3} ($Z_C = 2$) to 8.00×10^{-7} ($Z_C = 32$) at SNR= 5 dB under the AWGN channel. For Rayleigh Flat-Fading channel performance the selected values were ($Z_C = 2,4,8,16$), yielding ($N = 104,208,416,832$), at the same index $i_{LS} = 0$, as shown in Fig. 10.

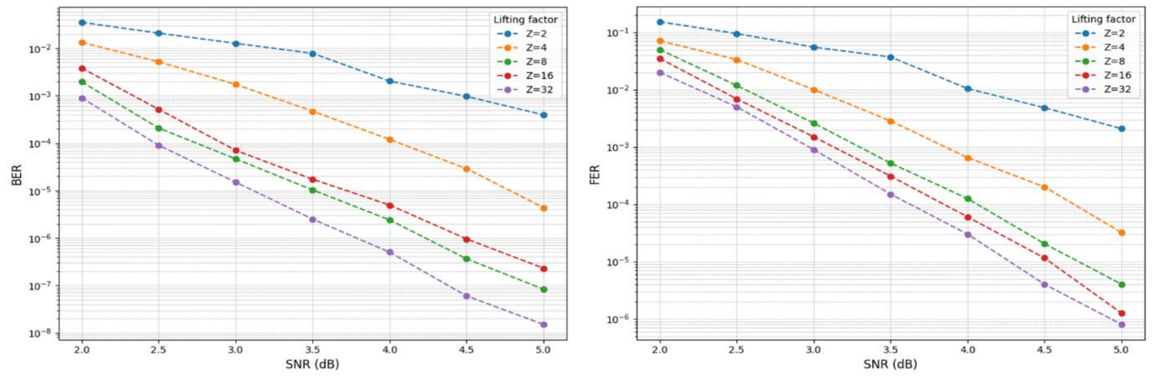


Figure 9. Comparison of lifting factor for AWGN channel for BER and FER

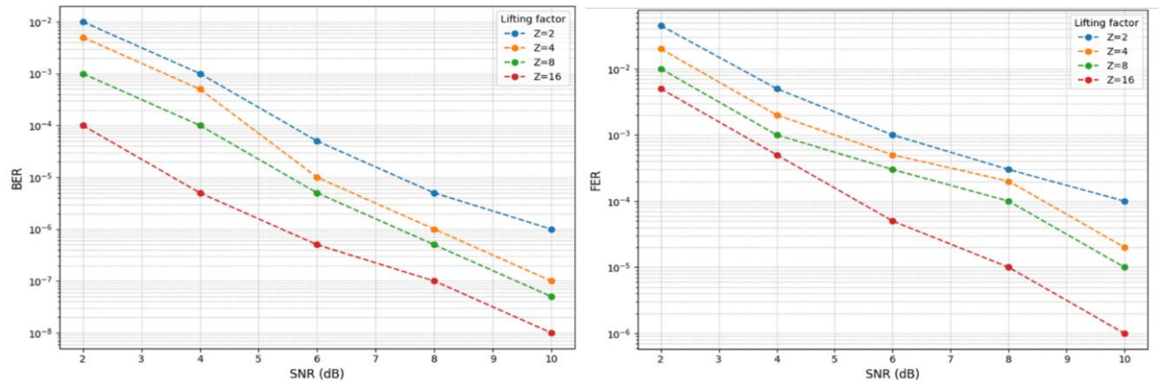


Figure 10. Comparison of lifting factor for Rayleigh fading channel for BER and FER

Channel-Scaling Coefficients

The impact of α on LDPC code performance is illustrated in Fig. with selected values (0.5, 0.75, 0.8, 0.9, and 1) under BG2, AWGN with R = 0.19, , $Z_C = 8$. A higher value of the scalar CN weight α accelerates convergence and produces reduced error rates, BER/FER, at low and moderate SNRs. The optimal performance occurs at α = 1, achieving BER=5.19e-07 and FER1.23e-05 at SNR=4.5 dB. Beyond approximately 3.5 dB, all configurations essentially reach the same error floor, so tuning α is most helpful when the channel is relatively noisy.

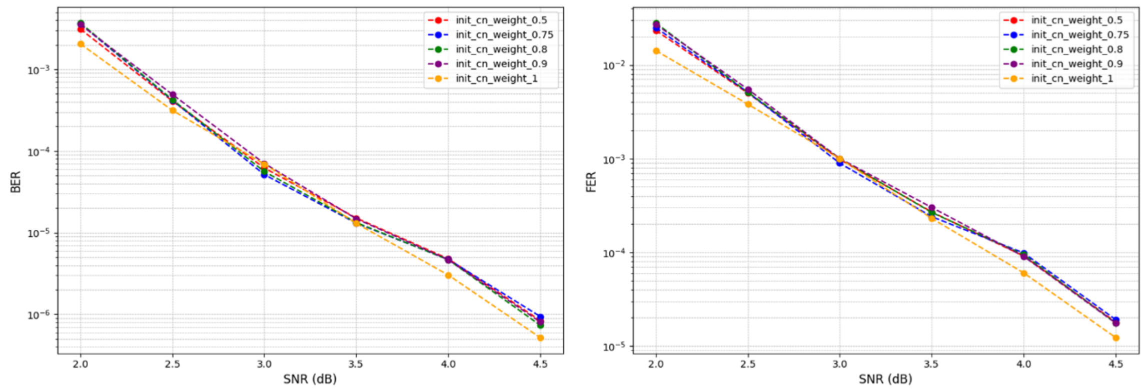


Figure 11. Comparison of check node for AWGN channel for BER and FER

Selection Criteria and Computational Overhead

Selection of uncorrected frames for which the syndrome check not equal to zero,as:

$$s = H \cdot \hat{x}^{\mathrm{T}} \neq 0$$

After maximum number of decoding iterations T=20 at SNR ≤ 2 dB is collected as “hard” samples for boosting. Fig. 12 displays the distribution of uncorrected frames over iterations. The number of uncorrected frames dropped from 5000 to 289 during the base decoder stage. At Base + Post, it was reduced to 2 frames, resulting in an improvement of 99%. The computational cost on an NVIDIA RTX 3090 GPU, the boosting stage adds approximately 15 % to the end-to-end training time

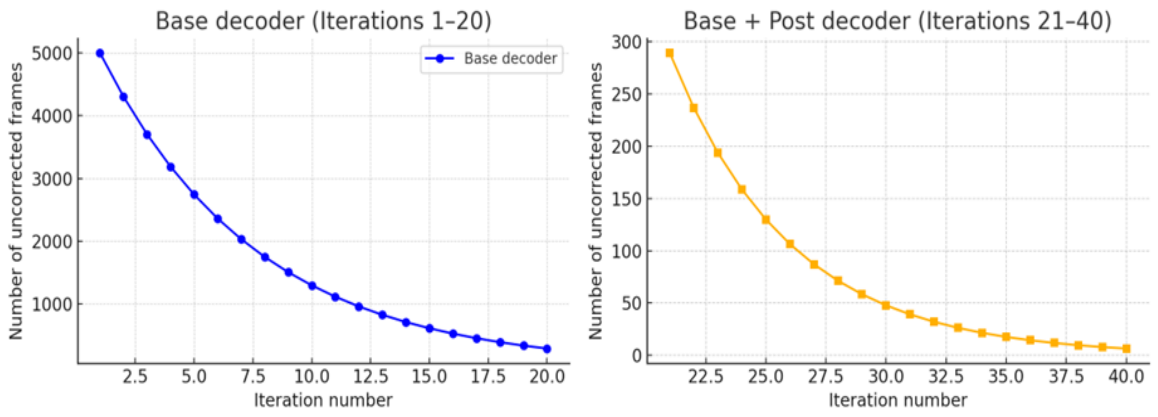


Figure 12. Distribution of uncorrected frames over iterations

Decoder Comparison

To explain the effectiveness of the proposed hybrid learning methodology, we compare decoding performance, BER, and FER using the AWGN channel for four cases: the MS, WMS, NMS Base decoder, and NMS Base + Post decoder, as shown in Fig. 13. The WMS decoder uses w(int-cn)=1 and w(int-ch)=1. While the WMS decoder exhibits better waterfall performance than the baseline MS decoder by 0.4 dB at 0.00001 BER, it still shows a significant error floor. The NMS Base decoder shows excellent performance in the waterfall region, similar to the performance curve of the WMS decoder, with a BER of 4.33e-06 and FER of 1.00e-04 at 4.5 dB. To improve both the waterfall region and error floor performance, the NMS Base + Post decoder, at SNR = 5 dB, achieves a BER of 1.42e-07 and FER of 3.18e-06, demonstrating superior performance with a 0.8 dB gain compared to the MS decoder.

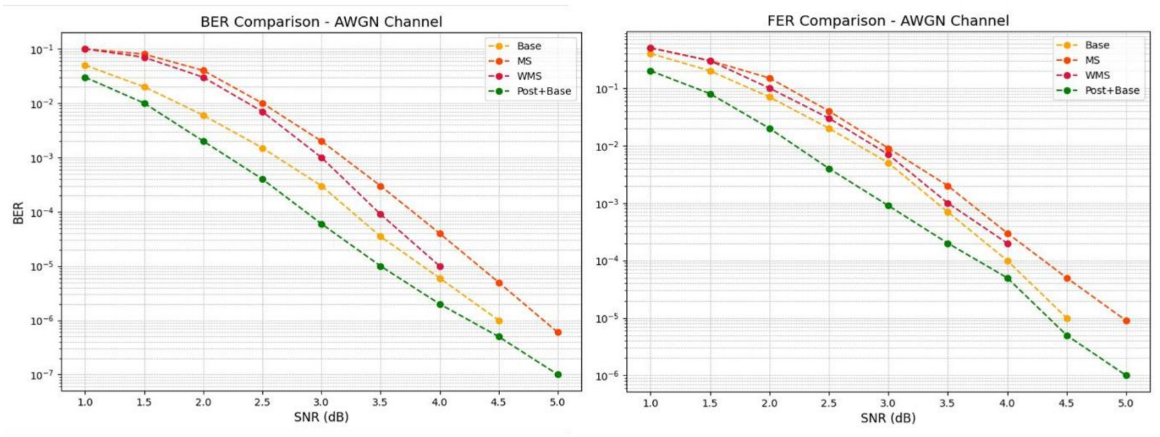


Figure 13. Comparison of decoder algorithm performance for BER and FER

Conclusion

In this study, the performance of the LDPC base-graph is improved in this work to meet to the requirements for 6G applications. We introduce a flexible and integrated two-stage adaptive neural decoding technique that incorporates a weighted Min-Sum (WMS) algorithm and syndrome-aware learning, utilizing parity-check feedback in real-time. In the first step, the Base decoder learns parameters including channel-scaling coefficients α end-to-end, per-iteration check-node weights$w_{ch}^{(t)}$, and, and per-edge variable-node parameters $w_{ij}^{vn},$ to jointly minimize loss functions, specifically a soft BER/FER. The second step, Base+Post, follows the same workflow and retrains the decoder using uncorrected frames to fine-tune residual correction weights, leading to a 0.83 dB waterfall improvement compared to traditional MS decoding at a 0.00001 BER value. The decoder optimizes its message-passing parameters based on error patterns and channel conditions. Various code rates (0.19 to 0.75) and lifting factors (Z = 2 to 32) were tested with different SNR values at a maximum of 20 iterations using QPSK schemes, along with comprehensive simulations that extend across AWGN and Rayleigh channels, demonstrating that BG1 outperforms BG2 in waterfalls and error floors. The adaptive WMS-NMS decoder offers an excellent solution for highly reliable, low-latency 6G scenarios, achieving BP-level accuracy with flexible channel adaptation and limited complexity.

Recommendations

According to the findings of our neural MS, this study offers the following recommendations:

1. Adaptation of dynamic scaling parameters: Adopt per-iteration learnable parameter check-node scaling factors (α) instead of a fixed value, while ensuring parameter sharing across iterations to reduce model complexity for the neural network.
2. Integration of hardware and real-time analysis: Evaluate real-time throughput, delay, and power usage by prototyping the proposed WMS decoder on FPGA or ASIC configurations to identify trade-offs between hardware complexity (latency, area, power) and decoding accuracy (error performance).
3. Modulation systems and models for wider networks: To assess efficiency in more challenging and practical wireless scenarios, examine the decoder's adaptability across higher-order modulations (16-QAM, 64-QAM) and various fading patterns (such as Nakagami-m and Rician).

Scientific Ethics Declaration

* The authors declare that the scientific, ethical, and legal responsibility of this article published in EPSTEM journal belongs to the authors.

Conflict of Interest

* The authors declare no conflict of interest.

Funding

* This work is unfunded.

Acknowledgements or Notes

* This article was presented as an oral presentation at the International Conference on Engineering and Advanced Technology (ICEAT) held in Selangor, Malaysia on July 23-24, 2025.

References

3GPP. (2020). *5G; NR; Multiplexing and channel coding Release 16*. TS 38.212 Version 16.2.0. Retrieved from https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

Cestari, R. G., Maroni, G., Cannelli, L., Piga, D., & Formentin, S. (2024). Split-boost neural networks. *IFAC-PapersOnLine*, *58*(15), 241–246.

Chen, J., Dholakia, A., Eleftheriou, E., Fossorier, M. P. C., & Hu, X. Y. (2005). Reduced-complexity decoding of LDPC codes. *IEEE Transactions on Communications*, *53*(8), 1288–1299.

Chung, S. Y., David Forney, G., Richardson, T. J., & Urbanke, R. (2001). On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. *IEEE Communications Letters*, *5*(2), 58–60.

Dai, J., Tan, K., Si, Z., Niu, K., Chen, M., Vincent Poor, H., & Cui, S. (2021). Learning to decode protograph LDPC codes. *IEEE Journal on Selected Areas in Communications*, *39*(7), 1983–1999.

Fossorier, M. P. C. (2001). Iterative reliability-based decoding of low-density parity check codes. *IEEE Journal on Selected Areas in Communications*, *19*(5), 908–917.

Kingma, D. P., & Ba, J. L. (2015). Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 1–15.

Kschischang, F. R., Frey, B. J., & Loeliger, H. A. (2002). Factor graphs and the sum-product algorithm. *IEEE Transactions on İnformation Theory, 47*(2), 498-519.Kumar, N., Kedia, D., & Purohit, G. (2023). A review of channel coding schemes in the 5G standard. *Telecommunication Systems*, *83*(4), 423–448.

Kwak, H.-Y., Yun, D.-Y., Kim, Y., Kim, S.-H., & No, J.-S. (2024). Boosted neural decoders: Achieving extreme reliability of LDPC codes for 6G networks. *arXiv 2405.13413*

Li, G., & Yu, X. (2023). The impact when neural min-sum variant meets ordered statistics decoding of LDPC codes. *arXiv preprint arXiv:2310.07129*.

Lian, M., Carpi, F., Hager, C., & Pfister, H. D. (2019, July). Learned belief-propagation decoding with simple scaling and SNR adaptation. *IEEE International Symposium on Information Theory - Proceedings*, 161–165.

Na, H., Park, H., Kwak, H. Y., & Ahn, S. K. (2025). Learning strategies for neural min-sum decoding of LDPC codes. *ICT Express*, *11*(1), 161–166.

Nachmani, E., Be'Ery, Y., & Burshtein, D. (2017). Learning to decode linear codes using deep learning. *54th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2016*, 341–346.

Nguyen, T. T. B., Tan, T. N., & Lee, H. (2019). Efficient QC-LDPC encoder for 5G new radio. *Electronics*, *8*(6), 668.

Petrović, V. L., El Mezeni, D. M., & Radošević, A. (2021). Flexible 5G new radio LDPC encoder optimized for high hardware usage efficiency. *Electronics*, *10*(9), 1106.

Richardson, T., & Kudekar, S. (2018). Design of low-density parity check codes for 5G new radio. *IEEE Communications Magazine*, *56*(3), 28–34.

Rowshan, M., Qiu, M., Xie, Y., Gu, X., & Yuan, J. (2024, February). Channel coding toward 6G: Technical overview and outlook. *IEEE Open Journal of the Communications Society*, *5*, 2585–2685.

Sun, K., & Jiang, M. (2018). A hybrid decoding algorithm for low-rate LDPC codes in 5G. *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*, 1–5.

Tanner, R. M. (1981). A recursive approach to low complexity codes. *IEEE Transactions on Information Theory*, *27*(5), 533–547.

TSGR. (2020). *TS 138 212 - V16.2.0 - 5G; NR; Multiplexing and channel coding (3GPP TS 38.212 version 16.2.0 Release 16)*. Retrieved from https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx

Wu, X., Song, Y., Jiang, M., & Zhao, C. (2010). Adaptive-normalized/offset min-sum algorithm. *IEEE Communications Letters*, *14*(7), 667–669.

Xiao, X., Raveendran, N., Vasic, B., Lin, S., & Tandon, R. (2021). FAID diversity via neural networks. *2021 11th International Symposium on Topics in Coding (ISTC)*, (pp. 1-5). IEEE.

Author(s) Information

Noor Salih Mohammed
Al-Furat Al-Al-Awsat Technical University, Najaf Technical College, Communications Engineering Department, Al-Najaf, Iraq.
e-mail: noor.saleh@student.atu.edu.iq

Ahmed Ghanim Wadday
Al-Furat Al-Al-Awsat Technical University, Najaf Technical College, Communications Engineering Department, Al-Najaf, Iraq.

Bashar Jabbar Hamza
Al-Furat Al-Al-Awsat Technical University, Najaf Technical College, Communications Engineering Department, Al-Najaf, Iraq.

To cite this article:

Mohammed, N.S., Wadday, A.G., & Hamza, B.J. (2025). Learning about syndrome awareness and WMS algorithm for adaptive neural decoding for 6G LDPC base graph enhancement. *The Eurasia Proceedings of Science, Technology, Engineering and Mathematics (EPSTEM), 37,* 574–589.