Deep Learning-Aided Belief Propagation Decoder for Polar Codes

Weihong Xu, *Student Member, IEEE*, Xiaosi Tan, *Member, IEEE*, Yair Be'ery, *Senior Member, IEEE*, Zaichen Zhang, *Senior Member, IEEE*, Xiaohu You, *Fellow, IEEE*, and Chuan Zhang, *Member, IEEE*

***Abstract*—This paper presents deep learning (DL) methods to optimize polar belief propagation (BP) decoding and concatenated LDPC-polar codes. First, two-dimensional offset Min-Sum (2-D OMS) decoding is proposed to improve the error-correction performance of existing normalized Min-Sum (NMS) decoding. Two optimization methods used in DL, namely back-propagation and stochastic gradient descent, are exploited to derive the parameters of proposed algorithms. Numerical results demonstrate that there is no performance gap between 2-D OMS and exact BP on various code lengths. Then the concatenated OMS algorithms with low complexity are presented for concatenated LDPC-polar codes. As a result, the optimized concatenated OMS decoding yields error-correction performance with CRC-aided successive cancellation list (CA-SCL) decoder of list size 2 on length-1024 polar codes. In addition, the efficient hardware architectures of scalable polar OMS decoder are described and the proposed decoder is reconfigurable to support three code lengths ($N = 256, 512, 1024$) and two decoding algorithms (2-D OMS and concat. OMS). The polar OMS decoder implemented on 65 nm CMOS technology achieves a maximum coded throughput of 5.4 Gb/s for code length 1024 and 7.5 Gb/s for code length 256, which are comparable to the state-of-the-art polar BP decoders. Moreover, a 5.1 Gb/s throughput is achieved under concat. OMS decoding mode for code length 1024 with a latency of 200 ns, which is superior to existing CA-SCL decoders that have similar error-correction performance.**

***Index Terms*—Polar codes, belief propagation (BP), deep learning, concatenated codes, ASIC implementation.**

I. Introduction

POLAR codes are regarded as a breakthrough of channel coding for their provably capacity-achieving capability over binary-input discrete memoryless channels (B-DMCs) [2]. Recently, increasing attention has been drawn towards polar codes since they are ratified as part of the 5G New Radio enhanced mobile broadband (eMBB) standard [3]. To meet the low-latency and high-speed requirements of 5G, lots of efforts have been made to design satisfactory polar decoders with good hardware efficiency [4]–[13].

Successive cancellation (SC) [2] and belief propagation (BP) [14] are two main decoding schemes for polar codes. SC is low-complexity and can achieve the channel capacity when the code length tends to infinity. However, SC is unable to provide satisfactory error-correction performance for moderate code lengths, and the decoding latency is high due to its sequential nature. The successive cancellation list (SCL) decoding [15] is proposed to improve the error-correction performance of SC. Moreover, SCL can be further enhanced by concatenating a cyclic redundancy check (CRC) code [15]. In [10]–[13], efficient implementations for the CRC-aided successive cancellation list (CA-SCL) decodings are studied from both the algorithm and the hardware perspectives, which attain promising error-correction performance.

Although optimized, SC or CA-SCL decoders [10]–[13] still suffer from limited throughput or high decoding latency. Compared to SC and CA-SCL, BP decoders [4]–[9] are more favorable for low-latency and high-throughput applications since they can process log-likelihood ratios (LLRs) in parallel. Pamuk [4] proposes the Min-Sum (MS) approximation that significantly reduces the decoding complexity of BP but deteriorates the performance in the meantime. Yuan *et al.* [16] add a normalization factor to alleviate the degradation of MS and show that normalization Min-Sum (NMS) has near-SC performance [5]. However, the normalization factors in [5, 16] are derived through brute-force search and confined to length-1024. Applying the normalization factors obtained in [5, 16] to longer codes, *e.g.* length $N = 4096$, will result in significant performance degradation compared to the exact BP (see Fig. 5). Besides, the complexity of brute-force search grows exponentially with the number of optimized parameters. There is no general method to determine the optimal parameter of NMS on arbitrary code lengths with acceptable complexity.

For low-density parity-check (LDPC) codes, density evolution (DE) [17, 18] is widely used to derive the parameters of modified BP decodings. However, DE is only suitable for the graphs that are loop-free or have large girth [17]. The short cycles in the factor graph of polar BP decodings [19] make DE not applicable for polar BP. Recently, Nachmani *et al.* [20] propose deep learning (DL) methods to optimize the graphs with short cycles (*e.g.* BCH codes) and achieve superior error-correction performance than the Sum-Product algorithm (SPA). For polar codes, Xu *et al.* [1] apply the DL methods on polar codes with length 64 and 512. But the drawback is the substantial amount of parameters. Meanwhile, the validity and effectiveness of DL methods have not been verified on longer codes. Therefore, the methods with higher efficiency are needed to optimize polar BP decoders.

On the other hand, there is still a huge error-correction performance gap between BP decoders in [4]–[9] and CA-SCL decoders in [10]–[13]. To close the gap, Guo *et al.* [21] concatenate a short LDPC code to polar codes. Abbas *et al.* [22] and Yu *et al.* [23] further improve the concatenated polar-LDPC codes by using the leaf set [19] and Bhattacharyya parameters

W. Xu, X. Tan, Z. Zhang, X. You, and C. Zhang are with the National Mobile Communications Research Laboratory, Southeast University, China. Email: chzhang@seu.edu.cn. *(Corresponding author: Chuan Zhang.)*

Y. Be'ery is with the School of Electrical Engineering, Tel-Aviv University, Tel-Aviv, 6997801 Israel, e-mail: ybeery@eng.tau.ac.il.

This paper was presented in part at IEEE International Workshop on Signal Processing Systems (SiPS), Lorient, France, 2017 [1].

[2] together to select intermediate channels. The performance gain of concatenated polar-LDPC codes is observed on various code lengths and code rates in [21, 23]. Due to the high parallel BP decodings of both polar and LDPC, the concatenated polar-LDPC decoder can yield high decoding throughput with a latency much lower than SC or CA-SCL. However, to the best of our knowledge, there is no efficient hardware implementation and optimization concerning concatenated polar-LDPC decoders.

In this paper, the design flow for polar BP decoder is presented from the algorithm level to the implementation level. Based on our previous work in [1], we propose the DL methods to optimize polar BP decodings in algorithm level. Then the co-design of algorithm and implementation is considered for the low-complexity hardware architectures. The contributions of this work include:

- The two-dimensional offset Min-Sum (2-D OMS) is proposed to improve the error-correction performance of modified BP decodings on various code lengths.
- The concatenated offset Min-Sum (concatenated OMS) is presented to decode the concatenated polar-LDPC codes with low complexity.
- The DL methods (back-propagation and gradient descent) are proposed to optimize the 2-D OMS and concatenated OMS decodings.
- A scalable OMS decoder, supporting both 2-D OMS and concat. OMS decodings, is presented to decode multi-length polar codes ($N = 256, 512$, and 1024).
- A pipelined schedule is shown to improve the hardware utilization and decoding throughput of proposed OMS decoder.

Numerical results show that there is no performance gap between the proposed 2-D OMS and exact BP on various code lengths $N = 256, 512, 1024, 4096$. Moreover, the proposed concatenated OMS decoder achieves comparable performance of CA-SCL decoder with list size 2. The proposed scalable polar OMS decoder is implemented with 65 nm 1P9M CMOS and integrates 1332k logic gates. With a power dissipation of 637 mW at 270 MHz, the proposed decoder achieves a coded throughput of 5.4 Gb/s on $N = 1024$ mode and 7.5 Gb/s on $N = 256$ mode. The resulting hardware and energy efficiency are comparative to the state-of-the-art (SOA) polar BP decoders [5]–[9]. Moreover, the proposed decoder can be configured to work under concat. OMS mode for $N = 1024$ and achieves 5.1 Gb/s throughput with a decoding latency of 54 cycles, which is much lower than the SOA CA-SCL decoders [10]–[13].

The remainder of this paper is organized as follows. Section II introduces polar codes and their decodings. Section III presents the 2-D OMS algorithms and the optimization methods based on DL. Section IV proposes the concatenated OMS decodings and the optimizations for concatenated polar-LDPC codes. The hardware architectures and quantization schemes for proposed decoding algorithms are presented in Section V. ASIC implementation results and comparison are given in Section VI. Finally, Section VII draws the conclusions.

A. Notation

Throughout this paper, a_k denotes the k-th entry of the vector $\mathbf{a}$. $|\mathcal{S}|$ denotes the cardinality of a countable set $\mathcal{S}$. Unless otherwise indicated, $\log(\cdot)$ and $\ln(\cdot)$ denote the base-2 and natural logarithm, respectively. A polar code with length N and code rate K/N is defined as $\mathcal{P}(N, K)$.

II. Preliminaries

A. Polar Codes

Polar codes are a type of linear block codes that exploit the channel polarization to achieve the capacity over the symmetric channel under long code lengths [2]. In the construction of a $\mathcal{P}(N, K)$ polar code, the N discrete channels are divided into reliable and unreliable ones by their capacities of correctly transmitting information. The K most reliable positions are defined as the information bits and $\mathcal{A}$ represents the set of indices of the information bit channels. The remaining $N - K$ bits are selected as the frozen bits and denoted by $\mathcal{A}^c$. The channel condition should be considered in the construction of polar codes. Throughout this paper, polar codes are constructed based on the Gaussian approximation (GA) [24].

The N-bit coded vector $\mathbf{x}$ can be generated by multiplying the information vector $\mathbf{u} = \{u_1, u_1, ..., u_N\}$ with the generator matrix $\mathbf{G}_N$ as $\mathbf{x} = \mathbf{u}\mathbf{G}_N = \mathbf{u}\mathbf{F}^{\otimes n}$, where $\mathbf{F}^{\otimes n}$ denotes the n-th *Kronecker* power of $\mathbf{F}$ and $n = \log N$, $\mathbf{F} = \left[\begin{smallmatrix} 1 & 0 \\ 1 & 1 \end{smallmatrix}\right]$. The encoding process can be efficiently realized with $\mathcal{O}(N \log N)$ exclusive or (XOR) complexity [2].

B. Belief Propagation Decoding

The BP decoding is realized through performing an iterative processing over the factor graph of any polar code $\mathcal{P}(N, K)$, where the factor graph is based on the corresponding generator matrix $\mathbf{G}_N$. In this case, the factor graph is composed of $n = \log N$ stages and $(n+1)N$ nodes in total. Each stage contains $N/2$ processing elements (PEs). Two types of LLRs, namely the left-to-right message $R_{i,j}^{(t)}$ and the right-to-left message $L_{i,j}^{(t)}$, are propagated iteratively over the factor graph, where i, j denotes the j-th node at the i-th stage and t denotes the t-th iteration. The right side of Fig. 1 gives an example of the factor graph for $\mathcal{P}(8, 4)$. Considering binary phase-shift keying (BPSK) modulation that maps $\mathbf{x} \in \{0, 1\}^N$ to $\mathbf{s} \in \{+1, -1\}^N$, the input LLRs of polar BP decoding are initialized as:

$$R_{1,j}^{(0)} = \begin{cases} 0, & \text{if } j \in \mathcal{A}, \\ +\infty, & \text{if } j \in \mathcal{A}^c, \end{cases} \quad L_{n+1,j}^{(0)} = \ln \frac{\Pr(x_j = +1 | r_j)}{\Pr(x_j = -1 | r_j)}, \quad (1)$$

where x_j and r_j denote the j-th bit of modulated and received codeword, respectively.

The PEs update and propagate the LLRs back and forth over the factor graph based on the iterative updating rules:

$$\begin{cases} L_{i,j}^{(t)} = \ g(L_{i+1,j}^{(t-1)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ L_{i,j+N/2^i}^{(t)} = \ g(L_{i+1,j}^{(t-1)}, R_{i,j}^{(t)}) + L_{i+1,j+N/2^i}^{(t-1)}, \\ R_{i+1,j}^{(t)} = \ g(R_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}), \\ R_{i+1,j+N/2^i}^{(t)} = \ g(R_{i,j}^{(t)}, L_{i+1,j}^{(t-1)}) + R_{i,j+N/2^i}^{(t)}, \end{cases} \quad (2)$$

where $g(x, y)$ is referred to as the box-plus operator:

$$g(x, y) \triangleq B(x \boxplus y) = \ln \frac{1 + e^{x+y}}{e^x + e^y}. \tag{3}$$

When the BP decoding reaches the maximum number of iterations T, the information bit $\hat{u}_j$ and transmitted codeword $\hat{x}_j$ are estimated based on their LLRs Λ_j^u and Λ_j^x, where $\Lambda_j^u = L_{1,j}^{(T)} + R_{1,j}^{(T)}$ and $\Lambda_j^x = L_{n+1,j}^{(T)} + R_{n+1,j}^{(T)}$. The hard decision is based on the following rules:

$$\hat{u}_j = \begin{cases} 0, & \text{if } \Lambda_j^u > 0, \\ 1, & \text{otherwise.} \end{cases} \tag{4}$$

C. Concatenated Polar-LDPC Codes

For finite-length polar codes, a proportion of information bit channels are not fully polarized (called intermediate channels), thus are not reliable enough to transmit information noiselessly. The intermediate bit channels degrade the error-correction performance of BP decoding. A short outer LDPC code is concatenated to the original polar codes to protect the intermediate channels in [21]. The concatenated polar-LDPC codes can be described as a 4-tuple $\mathcal{P}_{\text{concat}}(N, K, \mathring{N}, \mathring{K})$, where an outer $(\mathring{N}, \mathring{K})$ LDPC code with rate $\mathring{R} = \mathring{K}/\mathring{N}$ is adopted.

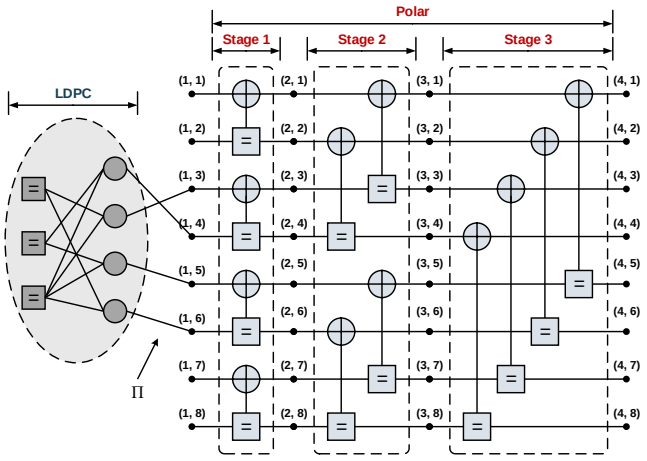


Fig. 1. Extended factor graph of concatenated polar-LDPC codes with $N = 8$ and $\mathring{N} = 4$.

The idea of concatenated polar-LDPC codes is to further divide the original information bit channels $\mathcal{A}$ into two types of subchannels: the good channels $\mathcal{A}_{\text{good}}$ and the intermediate channels $\mathcal{A}_{\text{inter}}$, where $\mathcal{A}_{\text{inter}} = \mathcal{A} \backslash \mathcal{A}_{\text{good}}$. A part of information bits concatenated with redundant parity-check bits are transmitted via intermediate channels $\mathcal{A}_{\text{inter}}$, whereas the other bits are directly transmitted through good channels $\mathcal{A}_{\text{good}}$. More precisely, $\mathring{K}$ bits of the total K-bit uncoded information data are first encoded by an LDPC encoder. Then the $\mathring{N}$-bit results are assigned to their corresponding intermediate channels $\mathcal{A}_{\text{inter}}$ according to the bit selection scheme defined by Π in Eq. (5) while the remaining $K - \mathring{K}$ information bits are assigned to good channels $\mathcal{A}_{\text{good}}$. The resulting $\mathbf{u}$ is finally processed by the polar encoder.

Similar to CA-SCL decoding, the construction of concatenated polar-LDPC codes is realized through adding extra non-frozen bits. For code $\mathcal{P}_{\text{concat}}(N, K, \mathring{N}, \mathring{K})$, a total of $K + (\mathring{N} - \mathring{K})$ best channels are allocated for $\mathcal{A}$, where $\mathring{N} - \mathring{K}$ is the redundant parity-check bits of outer LDPC code. Among $\mathcal{A}$, $K - \mathring{K}$ bits are assigned to good channels $\mathcal{A}_{\text{good}}$ while the remaining $\mathring{N}$ bits are for intermediate channels $\mathcal{A}_{\text{inter}}$. It should be noted that N and K denote the effective code length and information bits, respectively. Both N and K are identical with the original code $\mathcal{P}(N, K)$ and the effective code rate $R = K/N$ is unchanged.

The bit selection scheme Π defines a position mapping from LDPC $\mathcal{I}$ to intermediate channels $\mathcal{A}_{\text{inter}}$ as:

$$\Pi : \mathcal{I} \to \mathcal{A}_{\text{inter}}, \tag{5}$$

where $\mathcal{I} = \{i_1, i_2, ..., i_{\mathring{N}}\}, i_j = j$, denotes the position indices of LDPC code and $\mathcal{A}_{\text{inter}} = \{a_1, a_2, ..., a_{\mathring{N}}\}$ stores the corresponding intermediate channel indices. Fig. 1 illustrates an example of $\Pi = \{(1, 4), (2, 3), (3, 5), (4, 6)\}$. Once Π is determined, its inverse mapping can be decided as $\Pi^{-1} : \mathcal{A}_{\text{inter}} \to \mathcal{I}$. The detailed methods to derive the bit selection scheme Π are discussed in Section IV.

The decoding of concatenated polar-LDPC code is performed over the extended graph containing Tanner graph of LDPC codes and factor graph of polar codes. Fig. 1 illustrates the extended factor graph with $N = 8$ and $\mathring{N} = 4$. One additional BP iteration of LDPC codes is needed between the right-to-left and the left-to-right propagation of polar decoding.

D. Low-Complexity BP Decoding Algorithms

1) Min-Sum Decoding: Polar codes can be decoded by the original exact box-plus operation in Eq. (3), but the computational complexity of box-plus operation is high due to the logarithm and exponential functions. A single box-plus operation generally requires three additions, three exponentiations, one division, and one logarithm, which are impractical for efficient implementation. According to [17], The box-plus operation can be alternatively represented as follows:

$$B(x \boxplus y) = \text{sgn}(x)\text{sgn}(y) \min(|x|, |y|) + \ln \frac{1 + e^{-|x+y|}}{1 + e^{-|x-y|}}, \tag{6}$$

where $\text{sgn}(x)$ denotes the sign function. In order to reduce the decoding complexity, the logarithm term of Eq. (6) can be omitted. Thus, $g(x, y)$ in Eq. (3) is approximated by the Min-Sum (MS) decoding [4] as follows:

$$g(x, y) \approx \text{sgn}(x)\text{sgn}(y) \times \min(|x|, |y|). \tag{7}$$

2) Normalized Min-Sum Decoding: The MS decoding significantly reduces the arithmetic complexity by replacing the complex nonlinear computation with a simple comparison and two sign operations. However, the inaccurate computation of Eq. (7) causes decoding performance degradation. Normalized Min-Sum (NMS)[1] can help to improve the MS decoding by multiplying a normalization factor α [5, 16], and the updating function is given by:

$$g(x, y) \approx \alpha \cdot \text{sgn}(x)\text{sgn}(y) \times \min(|x|, |y|), \tag{8}$$

[1]Also called Scaled Min-Sum (SMS) in [5, 16].

where α is used to compensate the approximation error. The computational complexity of $g(x, y)$ is increased by one multiplication compared to MS decoding. For some specific α (*e.g.* $\alpha = 1 - 2^{-4} = 0.9375$), shift and addition can be utilized to simplify the multiplication operation.

III. Deep Learning Methods for 2-D Offset Min-Sum Decoding

In this section, the 2-D OMS algorithm is presented. Then DL-aided methods to obtain the parameters are introduced.

A. 2-D Offset Min-Sum Decoding

The bit-error rate (BER) performance of NMS with $\alpha = 0.9375$ has about 0.5 dB gain over MS decoding on $\mathcal{P}(1024, 512)$ code as observed in [5, 16]. One extra multiplication is needed to compute the $g(x, y)$ in Eq. (8). The other approach to compensate the degradation caused by MS approximation is offset Min-Sum (OMS) [17]:

$$g(x, y) \approx \mathrm{sgn}(x)\mathrm{sgn}(y) \times \max\Big(\min(|x|, |y|) - \beta, 0\Big), \quad (9)$$

where the $\max$ function is used to remove the contributions of LLRs whose magnitudes are smaller than offset factor β. Eq. (9) is more hardware-friendly compared with NMS since it only requires the sign, comparison, and subtraction operations. The $\max(x, 0)$ operation can be realized through one XOR gate. Thus the additional computational complexity is low.

The normalization factor α for NMS and offset factor β for OMS should vary for different iterations to obtain optimal performance [17]. The multiple-scaled MS decoding is presented in [1], where each node over the factor graph is parameterized by different normalization factors. However, the total number of required parameters is about $2N \log N$ for each iteration, which is impractical for efficient implementation. A balance between reduction of parameters and good performance is needed. For LDPC codes, Zhang *et al.* propose 2-dimensional normalization and 2-dimensional offset algorithms [18], where outgoing LLRs of both CNs and VNs are compensated with different factors. Similarly, Liao *et al.* [25] dynamically adjust the normalization factor at different decoding iterations to achieve near-optimal performance at the cost of little hardware overhead.

For polar BP decoding, the intermediate LLRs of left-to-right and right-to-left propagation in Eq. (2) have different dynamic ranges due to two reasons. First, the round-trip updating schedule [26] is performed sequentially between stages[2], which means the left-to-right LLRs $R_{i,j}^{(t)}$ are updated by the right-to-left LLRs $L_{i,j}^{(t)}$. Second, some left-to-right messages $R_{i,j}^{(t)}$ will tend to infinity since $R_{1,j}^{(0)}$ associated to frozen bits are initialized to infinity by Eq. (1) while all right-to-left messages $L_{i,j}^{(t)}$ will remain within a certain range. Hence, using one unified offset factor is not sufficient to compensate for the performance degradation. Inspired by [18, 25], two offset factors are used to parameterize the OMS function in Eq. (9) during left-to-right and right-to-left propagation, respectively.

[2]The decoding begins from right to left by default in this paper.

The two-dimensional offset Min-Sum (2-D OMS) decoding is expressed as the following equations:

$$\begin{cases} g_L(x, y) \approx \mathrm{sgn}(x) \cdot \mathrm{sgn}(y) \cdot \max\big(\min(|x|, |y|) - \beta_L, 0\big), \\ g_R(x, y) \approx \mathrm{sgn}(x) \cdot \mathrm{sgn}(y) \cdot \max\big(\min(|x|, |y|) - \beta_R, 0\big), \end{cases} \quad (10)$$

where the right-to-left and left-to-right approximate box-plus functions $g_L(x, y)$ and $g_R(x, y)$ are parameterized by right-to-left offset factor β_L and left-to-right offset factor β_R, respectively. The decoding complexity of the proposed 2-D OMS algorithm is the same with OMS. Moreover, 2-D OMS and OMS are identical when $\beta = \beta_L = \beta_R$. It can be expected that the performance of 2-D OMS decoding will be no worse than OMS if the parameters are properly selected.

B. Optimizing Polar BP Decoder through Training

2-D OMS algorithm offers one more dimension of freedom for the optimization compared to NMS and OMS with a single factor. However, it is difficult to obtain the optimal parameter combination through brute-force search as in [5, 16] since the parameter search space of 2-D OMS is a continuous and nonlinear minimization problem [18]. To this end, we investigate DL methods, namely back-propagation and mini-batch stochastic gradient descent (mini-batch SGD), to determine the good parameters of the modified BP decoding. Back-propagation [27] is a widely used learning procedure for training neural networks. Given a neural network with parameters to be trained and a loss function, the idea of back-propagation is to use the chain rule to calculate gradients of the loss function with respect to the training parameters. Then the error of the loss function can be reduced by adjusting the parameters based on gradient descent methods.

The methods above are not specified to the standard deep neural networks, and the fundamentals of back-propagation and SGD can also be utilized to optimize the variants of neural network which have different structures [28]. Our previous work [1] has demonstrated that the DL methods can efficiently optimize polar BP decoder even with large parameter space. We refine the basic definitions of DL aided optimization methods associated to polar BP decoding in this paper. The factor graphs of polar BP can be regarded as neural networks with special connection pattern and network structure. Without loss of generality, we consider the polar BP decoder as a function of received noisy codeword $\mathbf{r}$ and the parameter set $\boldsymbol{\theta}$, which is given as:

$$\hat{\mathbf{y}} = F(\mathbf{r}; \boldsymbol{\theta}), \quad (11)$$

where the decoder outputs soft LLRs $\hat{\mathbf{y}}$. When a noisy codeword vector $\mathbf{r}$ is fed and propagates through the network based on some decoding algorithms (NMS or OMS), the output $\hat{\mathbf{y}}$ will be finally produced. The parameter set $\boldsymbol{\theta}$ can be any one of α, β, or $\{\beta_L, \beta_R\}$.

Next, the loss function $\mathcal{L}$ is adopted to measure the decoding performance of $F(\mathbf{r}; \boldsymbol{\theta})$. Assuming that $\mathbf{y}$ is the desired output corresponding to input $\mathbf{r}$, the loss between the predicted output $\hat{\mathbf{y}}$ and the desired $\mathbf{y}$ over the given $\boldsymbol{\theta}$ is defined as $\tilde{J}(\boldsymbol{\theta}; \mathbf{r}, \mathbf{y}) = \mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = \mathcal{L}\left(\mathbf{y}, F(\mathbf{r}; \boldsymbol{\theta})\right)$, where the loss $\tilde{J}(\boldsymbol{\theta}; \mathbf{r}, \mathbf{y})$ is a scalar associated to $\boldsymbol{\theta}$, $\mathbf{r}$ and $\mathbf{y}$, similar to the BER metric. Back-propagation requires that the adopted

loss function $\mathcal{L}$ is differentiable. Hence, the BER can not be adopted as the loss function. In addition, the factors of NMS and OMS should satisfy $0 < \alpha \leq 1$ and $\beta > 0$ [25], respectively. The restriction of parameters can be realized through adding regularization penalty $\Omega(\boldsymbol{\theta})$ to $\tilde{J}(\boldsymbol{\theta}; \mathbf{r}, \mathbf{y})$:

$$J(\boldsymbol{\theta}; \mathbf{r}, \mathbf{y}) = \tilde{J}(\boldsymbol{\theta}; \mathbf{r}, \mathbf{y}) + \lambda \cdot \Omega(\boldsymbol{\theta}), \tag{12}$$

where $J(\boldsymbol{\theta}; \mathbf{r}, \mathbf{y})$ denotes the regularized loss and $\lambda, \lambda \geq 0$ controls the contribution of regularization term $\Omega(\boldsymbol{\theta})$ to the regularized loss function. The hyperparameter λ usually equals to a large positive value (*e.g.* 100) during training. The regularization functions for NMS and OMS (or 2-D OMS) are given by:

$$\Omega(\boldsymbol{\theta}) = \begin{cases} \sum\limits_{\theta \in \boldsymbol{\theta}} |\min(\theta, 0)| + \max(\theta, 1) - 1, & \text{NMS}, \\ \sum\limits_{\theta \in \boldsymbol{\theta}} |\min(\theta, 0)|, & \text{OMS}, \end{cases} \tag{13}$$

where the parameters out of the valid range will be penalized.

As the previous works [1, 20], binary cross entropy (BCE) loss in Eq. (14) is adopted. The soft output of polar BP decoder is in LLR domain and can not be processed by BCE loss since BCE loss accepts input in probability domain. Each LLR output of polar BP decoder should pass a sigmoid function $o_i = \sigma(\hat{y}_i) = (1 + e^{-\hat{y}_i})^{-1}, i \in \mathcal{A}$ to squash the LLRs within range $(-\infty, \infty)$ to probabilities within range $(0, 1)$. Then the following BCE loss function is applied:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{K} \sum_{i \in \mathcal{A}} \Big((1 - u_i) \log(o_i) + u_i \log(1 - o_i) \Big), \tag{14}$$

where u_i is the i-th bit of actual information vector $\mathbf{u}$. Polar BP decoding and AWGN channel both satisfy the symmetry properties, thus training using random codewords transmitted through the channel is equivalent to using the all-zero codewords. In the case of all-zero transmitted codewords, we have $u_i = 0$ and the BCE loss function in Eq. (14) is simplified as:

$$\mathcal{L}(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{K} \sum_{i \in \mathcal{A}} \log(o_i). \tag{15}$$

The iterative optimization method, stochastic gradient descent (SGD) [28], is used to search the good parameters. However, the plain SGD uses only one training sample to update the parameters at each training iteration, which may cause fluctuation. Alternatively, the mini-batch SGD is adopted to increase training stability. The training algorithm for polar BP decoding is summarized in Algorithm 1. At the beginning of each training iteration, a batch $\mathcal{B} = \{\mathbf{r}_1, ..., \mathbf{r}_M\}$ containing M training samples that are randomly sampled from channel output is fed into the polar BP decoder. Then the average regularized loss $J(\boldsymbol{\theta}^{(t)})$ is calculated by averaging the M losses. The new parameter set $\boldsymbol{\theta}^{(t+1)}$ is finally updated using the gradient of average loss $\frac{\partial J(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}^{(t)}} = \nabla J(\boldsymbol{\theta}^{(t)})$. The mini-batch SGD algorithm can be expressed as:

$$\begin{cases} J(\boldsymbol{\theta}^{(t)}) = \dfrac{1}{M} \sum\limits_{i=1}^{M} J(\boldsymbol{\theta}; \mathbf{r}_i, \mathbf{y}_i), \\ \boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \dfrac{\partial J(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}^{(t)}}, \end{cases} \tag{16}$$

Algorithm 1: Training algorithm for parameter set $\boldsymbol{\theta}$

Input : Initialized training parameters $\boldsymbol{\theta}^{(1)}$, and maximum training epochs N_E.
Output : The trained parameter $\boldsymbol{\theta}^*$.

for $t = 1, 2, ..., N_E$ **do**
  Sample a batch of codewords $\mathcal{B} = \{\mathbf{r}_1, ..., \mathbf{r}_M\}$;
  for $i = 1, 2, ..., M$ **do**
    // Polar decoding
    $\hat{\mathbf{y}}_i \leftarrow F(\mathbf{r}_i; \boldsymbol{\theta}^{(t)})$;
  // Computing average loss
  $J(\boldsymbol{\theta}^{(t)}) \leftarrow \frac{1}{M} \sum_{i=1}^{M} J(\boldsymbol{\theta}^{(t)}; \mathbf{r}_i, \mathbf{y}_i)$;
  // Computing gradient
  $\nabla J(\boldsymbol{\theta}^{(t)}) \leftarrow \frac{\partial J(\boldsymbol{\theta}^{(t)})}{\partial \boldsymbol{\theta}^{(t)}}$;
  // Updating parameters
  $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \eta_t \nabla J(\boldsymbol{\theta}^{(t)})$;
$t^* \leftarrow \arg\min_{t \in N_E} J(\boldsymbol{\theta}^{(t)})$;
return $\boldsymbol{\theta}^{(t^*)}$;

where M is the mini-batch size and η_t is the learning rate at the t-th iteration. $\mathbf{y}_i$ is the all-zero information vector associated to the i-th LLR output of the training batch $\mathcal{B}$.

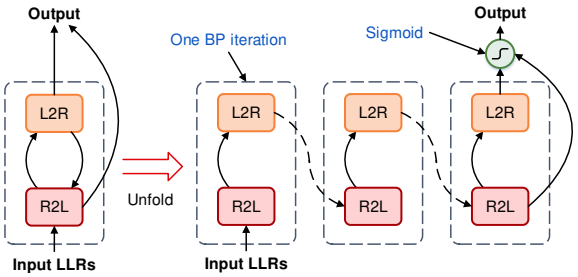


Fig. 2. Feed-forward architectures of unfolded polar BP decoder (equivalent to 3 full BP iterations).

The iterative polar BP decoder has similar structures with the recurrent neural networks. For the convenience of training, the repetitive architecture of polar BP decoder can be unfolded and represented by the feed-forward architectures as Fig. 2, where one time step consists of a right-to-left propagation (R2L) and a left-to-right propagation (L2R), equivalent to one BP iteration. The parameters $\boldsymbol{\theta}$ are shared with different iterations. Each R2L or L2R corresponds to $n = \log N$ layers, respectively. Hence, each time step has $2n$ layers while the last time step has $2n+1$ after adding a sigmoid layer. The constructed feed-forward polar BP decoder can be optimized by Algorithm 1.

C. Numerical Results

We apply the aforementioned DL methods to optimize NMS and 2-D OMS decoders. Computing the gradient manually for such networks is intractable, therefore the DL library Tensorflow [29] is exploited to calculate the gradient automatically. In [30], it is proved that the optimal training SNRs for Additive White Gaussian Noise (AWGN) channel are around 1 dB and 2 dB. As a result, each mini-batch of the training sets is composed of $M = 400$ zero codewords transmitted through AWGN channel with $E_b/N_0 = 1$ dB and 2 dB, where 200 codewords are sampled at each SNR. The optimization method

Adam [31] with initial learning rate 0.01 is applied to adaptively tune the learning rate and accelerate the convergence. The polar BP decoder with 5 iterations is unfolded into feed-forward architectures. Such iteration number is sufficient for training because the approximation accuracy of the first few iterations dominates the overall error-correction performance according to [25]. Besides, more unfolding iterations will increase the network depth, where the vanishing gradient problem [28] will occur, thus making the parameters hard to optimize. Unless explicitly specified, the parameter set $\boldsymbol{\theta} = \{\alpha\}$ for NMS is initialized with one. For 2-D OMS, the parameter set $\boldsymbol{\theta} = \{\beta_L, \beta_R\}$ are initialized with all-zeros.

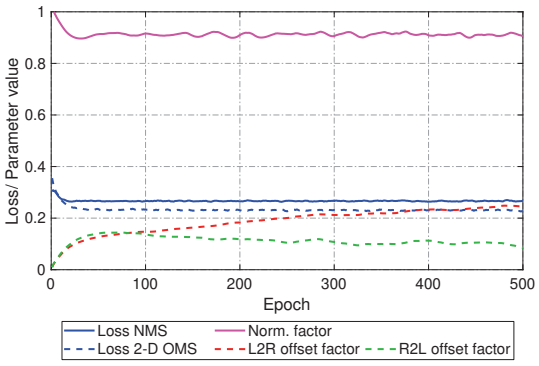


Fig. 3. Evolution of training loss and trained parameters (normalization factor α and offset factors $\{\beta_L, \beta_R\}$) for $\mathcal{P}(1024, 512)$.

In the first experiment, NMS and 2-D OMS decoders for $\mathcal{P}(1024, 512)$ are trained, respectively. Fig. 3 illustrates the evolution of BCE training loss and trained normalization factor α and offset factors $\{\beta_L, \beta_R\}$. The trend of parameters is stable after 500 epochs, indicating that the training process has converged. It is observed that the α resulting in the minimum loss value is around 0.92, very close to the empirical value $\alpha = 0.9375$ in [16] through brute-force search. The offset factors resulting in the minimum loss is $\boldsymbol{\theta} = \{\beta_L, \beta_R\} = \{0.08, 0.25\}$.

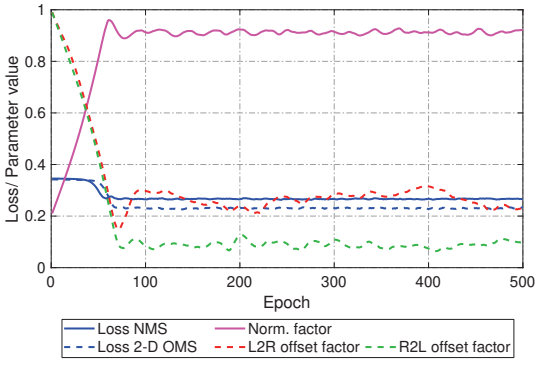


Fig. 4. Evolution of training loss and trained parameters (normalization factor α and offset factors $\{\beta_L, \beta_R\}$) for $\mathcal{P}(1024, 512)$ with different initializations.

To evaluate the robustness of proposed DL methods, we train NMS and 2-D OMS decoders with different parameter initializations such that the normalization factor and two offset factors are initialized with $\boldsymbol{\theta}^{(1)} = \{0.2\}$ and $\boldsymbol{\theta}^{(1)} = \{1.0, 1.0\}$, respectively. The training results are shown in Fig. 4. Even starting with different points, the training parameters can also converge to the similar results of Fig. 3, indicating that the proposed optimization methods are not sensitive to the initialization conditions.

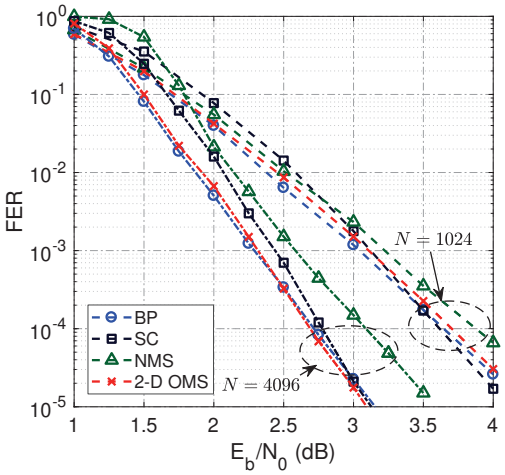


Fig. 5. Performance comparison for various BP decoding algorithms ($T_{\max} = 60$) and SC decoding on $\mathcal{P}(1024, 512)$ and $\mathcal{P}(4096, 2048)$.

The 2-D OMS decoder is also trained on longer code $\mathcal{P}(4096, 2048)$. The resulted parameters are $\boldsymbol{\theta} = \{\beta_L, \beta_R\} = \{0.14, 0.24\}$. The frame-error rate (FER) performance of BP, MS, NMS, and 2-D OMS on different code lengths is presented in Fig. 5. BP denotes the decoder with exact LLR updating in Eq. (3). For the fair comparison, the normalization factor of NMS for $\mathcal{P}(1024, 512)$ and $\mathcal{P}(4096, 2048)$ is $\alpha = 0.9375$, identical with [5, 7, 16]. The 2-D OMS uses the trained offset factors as mentioned above. It is observed that for $\mathcal{P}(1024, 512)$ the 2-D OMS outperforms NMS by 0.1 to 0.2 dB when $E_b/N_0 > 2.5$ dB. The NMS of $\mathcal{P}(4096, 2048)$ introduces 0.3 to 0.5 dB degradation compared to the exact BP. Instead, the proposed 2-D OMS decodings achieve comparable performance with exact BP on both two code lengths. The results above demonstrate that the proposed DL methods can effectively optimize the parameters for the variants of MS decoding on different code lengths.

IV. Offset Min-Sum Decoding for Concatenated Polar-LDPC Codes

In this section, low-complexity decoding algorithms and corresponding optimization methods are presented for concatenated polar-LDPC codes.

A. Concatenated Offset Min-Sum Decoding

In [21], the concatenated polar-LDPC codes are decoded using the exact LLR updating functions with high arithmetic complexity. Abbas *et al.* [22] use reduced-complexity NMS algorithms with $\alpha = 0.9375$ to decode both polar and LDPC codes. However, the inaccurate LLR updating of NMS will lead to performance degradation in high SNR region according to our simulation in Section IV-D. As in [17, 18],

Algorithm 2: Concatenated OMS decoding algorithm for concatenated polar-LDPC codes $\mathcal{P}_{\text{concat}}(N, K, \mathring{N}, \mathring{K})$

Input : Initialized LLRs $L_{n+1,j}^{(0)}$, frozen bit channels $\mathcal{A}^c$, bit selection scheme Π, and maximum iterations $T_{\max}$.
Output : Decoded soft output $\mathbf{\Lambda}^u$ and bits $\hat{\mathbf{u}}$.

// Initialization
foreach $j \in \mathcal{A}^c$ **do**
  $R_{1,j}^{(0)} = +\infty$;
for $t = 1, 2, ..., T_{\max}$ **do**
  // Right-to-left propagation
  for $i = \log N, \log N - 1, ..., 1$ **do**
    Update all $L_{i,j}^{(t)}$ based on Eq. (2) and Eq. (10);
  // BP iteration of LDPC
  Initialize $V_n^{(0)}$ from $L_{1,j}^{(t)}$ by Π;
  Update $U_{m\to n}^{(1)}$ based on Eq. (17);
  Update $V_n^{(1)}$ based on the extrinsic LLRs;
  Feed $V_n^{(1)}$ back to $R_{1,j}^{(t)}$ by Π^{-1};
  // Left-to-right propagation
  for $i = 1, 2, ..., \log N$ **do**
    Update all $R_{i+1,j}^{(t)}$ based on Eq. (2) and Eq. (10);
  Make hard decision on $\hat{\mathbf{u}}$ and $\hat{\mathbf{x}}$;
  // Early termination check
  if $t == T_{\max}$ **or** $\hat{\mathbf{x}} == \hat{\mathbf{u}}\mathbf{G}$ **then**
    return $\mathbf{\Lambda}^u$, $\hat{\mathbf{u}}$;

Algorithm 3: Bit selection scheme Π for LDPC codes

Input : $\mathcal{P}_{\text{concat}}(N, K, \mathring{N}, \mathring{K})$, and σ.
Output : Bit selectioin scheme Π.
1 $\mathbb{E}[W_1^1] \leftarrow 2/\sigma^2$, $n \leftarrow \log N$, $|\mathcal{A}| \leftarrow K + \mathring{N} - \mathring{K}$, and $\mathcal{A}_{\text{inter}} \leftarrow \{\}$;
// Gaussian approximation
2 **for** $j = 1, 2, ..., n$ **do**
3 **for** $i = 1, 2, ..., 2^j$ **do**
4 $\mathbb{E}[W_{2i-1}^{2^j}] \leftarrow \phi^{-1}\left(1 - \left(1 - \phi(\mathbb{E}[W_i^{2^{j-1}}])\right)^2\right)$;
5 $\mathbb{E}[W_{2i}^{2^j}] \leftarrow 2\mathbb{E}[W_i^{2^{j-1}}]$;
6 Sort $\mathbb{E}[W_i^N]$ in descending order and select the first $|\mathcal{A}|$ channel indices as $\mathcal{A}$;
// Bit selection by the size of leaf set
7 Compute the row weight $\boldsymbol{\mathcal{W}}$ of $\mathbf{G}_N$;
8 Divide $\mathcal{A}$ into k subsets $\mathcal{A}_1, \mathcal{A}_2, ..., \mathcal{A}_k$, where $\forall a_i, a_{i+1} \in \mathcal{A}_l$, $\mathcal{W}_{a_i} = \mathcal{W}_{a_{i+1}}$, $\mathbb{E}[W_{a_i}^N] < \mathbb{E}[W_{a_{i+1}}^N]$ and $\forall a_i \in \mathcal{A}_l$, $a_j \in \mathcal{A}_{l+1}$, $\mathcal{W}_{a_i} < \mathcal{W}_{a_j}$;
9 **for** $i = 1, 2, ..., k$ **do**
10 **if** $|\mathcal{A}_{inter}| + |\mathcal{A}_i| \leq \mathring{N}$ **then**
11 $\mathcal{A}_{\text{inter}} \leftarrow \mathcal{A}_{\text{inter}} \cup \mathcal{A}_i$;
12 **else**
13 **for** $j = 1, 2, ..., \mathring{N} - |\Pi|$ *and* $a_j \in \mathcal{A}_i$ **do**
14 $\mathcal{A}_{\text{inter}} \leftarrow \mathcal{A}_{\text{inter}} \cup \{a_j\}$;
15 Construct Π based on Eq. (5);
16 **return** Π;

the updating function for CNs is simplified as the following OMS approximation:

$$U_{m\to n}^{(t)} \approx \prod_{n' \in \mathcal{N}(m)\backslash n} \text{sign}(V_{m\to n'}^{(t-1)}) \times \max\left(\min_{n' \in \mathcal{N}(m)\backslash n} \left|V_{m\to n'}^{(t-1)}\right| - \beta_{\text{LDPC}}, 0\right), \tag{17}$$

where β_{LDPC} is the offset factor to compensate the BP decoding of outer LDPC codes. After considering the 2-D OMS decoding for polar codes, the complete concatenated OMS decoding for concatenated polar-LDPC codes is presented in Algorithm 2. A total of three offset factors, β_L, β_R, and β_{LDPC}, are used in concatenated OMS decoding. The early termination check uses the G-matrix criterion in [5].

B. Bit Selection Scheme for Outer LDPC Codes

Given $\mathcal{P}_{\text{concat}}(N, K, \mathring{N}, \mathring{K})$, the other question is how to select the $\mathring{N}$ intermediate channels $\mathcal{A}_{\text{inter}}$ from the $K + \mathring{N} - \mathring{K}$ non-frozen channels. Guo *et al.* [21] use the Bhattacharyya parameter [2] of each channel, $Z_i, i \in \mathcal{A}$, to determine the good and intermediate channels. The channels with Bhattacharyya parameter smaller than a threshold are considered as good channels $\mathcal{A}_{\text{good}} = \{a_i : Z_{a_i} < \xi_1\}$, whereas the channels satisfy $\xi_1 \leq Z_{a_i} < \xi_2$ are defined as intermediate channels $\mathcal{A}_{\text{inter}}$. However, Eslami *et al.* in [19] show that the size of leaf set can be a better metric to evaluate the reliability for each channel under BP decoding such that the channels with larger size of leaf set have higher reliability. Combining Bhattacharyya parameters, the authors of [22, 23] determine the intermediate channels through exploiting the size of leaf set.

For the channel construction of plain polar codes $\mathcal{P}(N, K)$, GA-based methods [24] estimate the mean value of LLRs rather than Bhattacharyya parameters [2] for each channel. The channels with larger mean LLRs have higher reliabilities. In this case, we utilize the leaf set in [19] together with metrics of GA [24] to determine the set of intermediate channels $\mathcal{A}_{\text{inter}}$. Then the bit selection scheme Π can be uniquely determined according to Eq. (5). The steps to derive Π are presented in Algorithm 3, where $\mathbb{E}(W_i^N)$ denotes the mean value of LLRs for the i-th channel W_i^N with code length N and the function ϕ is defined in [24]. First, under the assumption that all-zero codeword is transmitted, the LLR value of channel W_1^1 follows a normal distribution with mean $2/\sigma^2$ and variance $4/\sigma^2$ given the standard deviation σ of targeting AWGN channel [24]. The mean value of LLRs $\mathbb{E}(W_i^N)$ for a channel of length N can be obtained through recursive GA and the elements of $\mathcal{A}$ are selected based on these values. Since the size of leaf set is equivalent to the associated row weight of generator matrix $\mathbf{G}_N$ [19], the channels in $\mathcal{A}$ with the least weights are selected as the $\mathcal{A}_{\text{inter}}$ as shown from line 8 to line 16 of Algorithm 3.

C. Optimization of Concatenated Polar-LDPC Codes

The proposed concatenated OMS decoder can also be optimized by the DL methods in Section III-B. Compared to plain polar BP decoder, one iteration of LDPC OMS decoding with feed-forward architectures should be added between R2L and L2R modules in Fig. 2. Nachmani *et al.* show that the Tanner graphs can be unfolded into feed-forward network structure [20, Fig. 1]. We use similar methods to construct the feed-forward OMS decoder for outer LDPC codes.

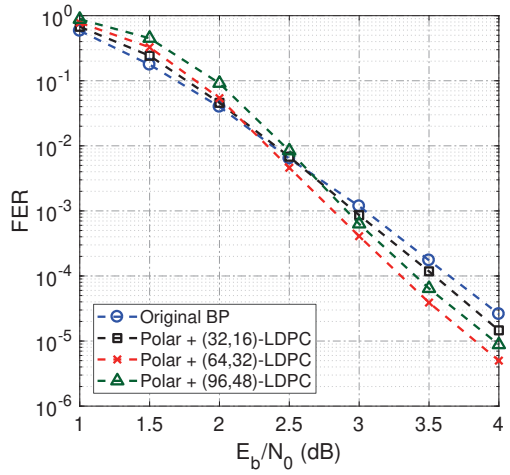


Fig. 6. Performance comparison between different concatenation schemes and original BP on polar codes $N = 1024$ and $R = 1/2$. ($T_{\max} = 60$)

The parameters of LDPC code in $\mathcal{P}_{\text{concat}}(N, K, \mathring{N}, \mathring{K})$ will significantly affect the error-correction performance. To the best of our knowledge, there is no theoretical study to determine the optimal code length and construction methods for the concatenated outer LDPC code. We fix the polar code with $N = 1024$ and $R = 1/2$ and then test three different short $(3, 6)$-regular LDPC codes ($\mathring{N} = 32, 64, 96$) constructed by progressive edge-growth [32]. Fig. 6 illustrates the performance comparison between three concatenated codes with the original polar BP. All of the codes are decoded by the exact BP algorithms. It can be seen that the gain provided by short LDPC code ($\mathring{N} = 32$) is smaller than the other two codes. Also, the LDPC code $\mathring{N} = 96$ leads to more degradation for $E_b/N_0 < 2.5$ dB and less performance gain for $E_b/N_0 > 2.5$ compared to $\mathring{N} = 64$. Consequently, $\mathcal{P}_{\text{concat}}(1024, 512, 64, 32)$ is the most suitable configurations for $N = 1024$ polar code among the three candidates.

D. Numerical Results

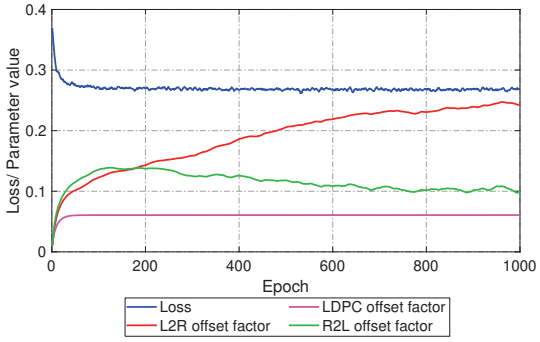


Fig. 7. Evolution of training loss and trained offset factors (β_L, β_R, and β_{LDPC}) for $\mathcal{P}_{\text{concat}}(1024, 512, 64, 32)$.

The DL methods in Section IV-C are used to optimize the concatenated OMS decoder for $\mathcal{P}_{\text{concat}}(1024, 512, 64, 32)$. The training parameters $\boldsymbol{\theta} = \{\beta_L, \beta_R, \beta_{\text{LDPC}}\}$ are initialized with all zeros. The other configurations are identical with 2-D OMS in Section III-C. As shown in Fig. 7, it takes about 1000 epochs to stabilize the training process and the trained offset factors are finally $\boldsymbol{\theta} = \{\beta_L, \beta_R, \beta_{\text{LDPC}}\} = \{0.1, 0.23, 0.06\}$.

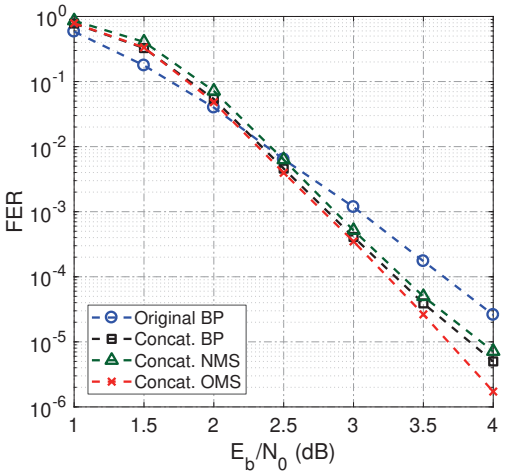


Fig. 8. Performance comparison for different decoding algorithms ($T_{\max} = 60$) for $\mathcal{P}_{\text{concat}}(1024, 512, 64, 32)$.

Fig. 8 shows the performance comparison between original BP decoding and variants of concatenated decoding algorithms for $\mathcal{P}_{\text{concat}}(1024, 512, 64, 32)$. The three offset factors of concatenated OMS decoder are $\{\beta_L, \beta_R, \beta_{\text{LDPC}}\} = \{0.1, 0.23, 0.06\}$. The normalization factor of concatenated NMS is $\alpha = 0.9375$ as in [22]. The concatenated OMS decoding slightly outperforms concatenated BP by 0.2 dB when $E_b/N_0 > 3$ dB and shows at most 0.5 dB improvement over original BP. It should be noted that although we only test on $N = 1024, R = 1/2$ polar code, the concatenated polar-LDPC codes are verified to obtain performance gain on other code lengths or code rates such as $N = 512$ in [23], and $N = 4096$ in [21, 23].

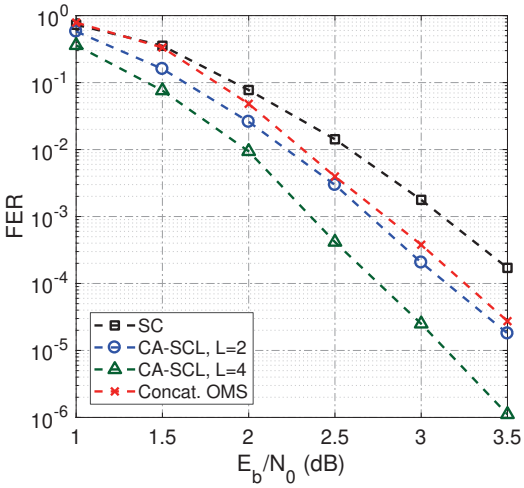


Fig. 9. Performance comparison for SC, CA-SCL, and concatenated OMS ($T_{\max} = 60$) decodings on polar codes $N = 1024$ and $R = 1/2$.

In Fig. 9, we also compare the proposed concatenated OMS algorithms with other decoding methods, namely SC and CA-SCL ($L = 2, 4$). The CRC length for CA-SCL decodings is

16. Concatenated OMS outperforms SC decoding and achieves comparable performance with CA-SCL of list size $L = 2$. There is a 0.3 to 0.5 dB gap between concatenated OMS and CA-SCL with list size $L = 4$. However, the advantage of concatenated OMS is that it can be implemented with much higher parallelism than CA-SCL.

V. Proposed Scalable Polar OMS Decoder

In this section, the hardware architectures for scalable polar OMS decoder are presented.

A. Overview of Hardware Architectures

The overall hardware architectures of proposed scalable polar OMS decoder are shown in Fig. 10. The system consists of five main parts: 1) double-column polar decoder [6, 33], 2) LDPC decoder, 3) termination unit, 3) memory and buffers, and 4) control logic. The scalable polar OMS decoder is reconfigurable to work under four decoding modes and support three code lengths: 1) 2-D OMS decoder for $N = 256$, 2) 2-D OMS decoder for $N = 512$, 3) 2-D OMS decoder for $N = 1024$, and 4) concat. OMS decoder for $N = 1024$. A maximum of 4 frames and 2 frames data can be processed in parallel for code length $N = 256$ and $N = 512$, respectively. The detailed designs are introduced in the following sections.

B. Quantization

The trained offset factors in Section III-C and Section IV-D are quantized to fixed-point format with 2 fractional bits, where the quantized offset factors become $\{\beta_L, \beta_R\} = \{0.0, 0.25\}$ for 2-D OMS and $\{\beta_L, \beta_R, \beta_{\text{LDPC}}\} = \{0.0, 0.25, 0.0\}$ for concatenated OMS. Considering that LDPC BP is not sensitive (6-bit in [25]) to quantization as polar BP (7-bit in [5]), the quantization bits Q_{LDPC} of LDPC decoder and Q_{polar} of polar decoder are separately determined. The configuration of $Q_{\text{LDPC}} = 6$ and $Q_{\text{polar}} = 7$ is used. The FER performance comparison for 2-D OMS and concatenated OMS decodings under different code lengths is shown in Fig. 11. The floating point with maximum iterations $T_{\max} = 60$ is taken as the baseline. $T_{\max}$ of quantized 2-D OMS decodings is set to 20 and $T_{\max}$ of concat. OMS decoding is set to 30. It can be seen that the above configurations are able to provide error-correction performance with negligible degradation.

C. Processing Element (PE) Array

Two PE arrays are employed in the double-column decoder [6, 33] to update the LLRs of two consecutive stages over the factor graph. Each PE array with $N/2$ PEs (see Fig. 12(a)) implements 2-D OMS algorithms in Eq. (10). Each PE calculates $x = g(b, c + d)$ and $y = g(a, b) + c$ in one direction, consisting of two adders and two g units. Fig. 12(b) shows the OMS-based g function unit in PE. The compare and select unit calculates the minimum absolute value of two inputs. Then the result is subtracted by offset factors and compare with zero. Since the right-to-left and the left-to-right propagations have different offset factors, namely $\{\beta_L, \beta_R\} = \{0.0, 0.25\}$, the offset unit in g unit is controlled by the L/R signal to switch for different offset factors.

D. LDPC Decoder

The LDPC decoder has a fully parallel structure (see Fig. 13) which can process the incoming LLRs in one clock cycle. The bit-select unit selects and latches $\mathring{N}$ LLRs from $L_{1,j}^{(t)}$ according to the mapping Π whereas the bit-expand unit feeds the decoding results $R_{1,j}^{(t)}$ back to the right memory and the termination unit according to Π^{-1} and $\mathcal{A}^c$. The check node processor (CNP) containing 32 CNs processes latched LLRs from the bit-select unit. The variable node processor (VNP) with 64 VNs calculates the sum of extrinsic messages. The CNP and VNP are equivalent to an MS decoder since the offset of LDPC decoder is quantized to zero in Section V-B.

E. Scalable Designs for Multi-length Polar Codes

For polar BP decoder, the difference of polar codes with different rates is the positions of left-most left-to-right LLRs $R_{1,j}^{(0)}$ that are initialized to ∞. Therefore, the polar BP decoder can support multi-rate codes simply through initializing different $R_{1,j}^{(0)}$ according to the frozen bit positions. However, the existing polar BP decoders [5]–[9] only support polar codes with fixed code length. The following scalable designs are presented to address this problem.

1) Flexible Routing Networks: For an N-length decoder, the PE and memory resources are sufficient to decode polar codes with shorter lengths due to the recursive properties of factor graph. More specifically, the factor graph of polar code with length N can be divided into two $N/2$-subgraphs or four $N/4$-subgraphs as shown in Fig. 14. Thus the decodings of $N/2$ and $N/4$ codes can be performed on the factor graph of N-length polar code by activating only $\log(N/2)$ and $\log(N/4)$ stages of PEs, respectively. Besides, the activated number of PEs in each stage is only $N/4$ and $N/8$ for $N/2$ and $N/4$ code lengths, respectively. In this case, 2 frames of $N/2$ codes or 4 frames of $N/4$ codes can be concurrently processed by $N/2$ PEs within one clock cycle. The key difference between polar codes with different lengths is the interconnections of two consecutive stages as illustrated in the right side of Fig. 14. The scalable polar OMS decoder of this work for multi-length codes is achieved by implementing 3 types of routings ($N = 1024, 512, 256$) in the routing networks of the double-column polar decoder. A control signal is used to select the routings under different decoding modes.

2) Memory Hierarchy: The register-based memory is divided into two blocks, the left memory and the right memory. The left memory stores the intermediate LLRs of even columns whereas the right memory stores the LLRs of odd columns over the factor graph. Each memory block is cascaded by 4 banks of $(NQ_{\text{polar}}/4)$-bit registers and the depth of each bank is $\log N/2$ words. The banks of each memory block are controlled by separated signals to support the memory access of three code lengths $N = 1024, 512, 256$. Besides, the memory blocks are pipelined to reduce the critical path.

F. Termination Unit

The iterations are redundant after the decoding process has converged. Several criteria can be exploited to improve the

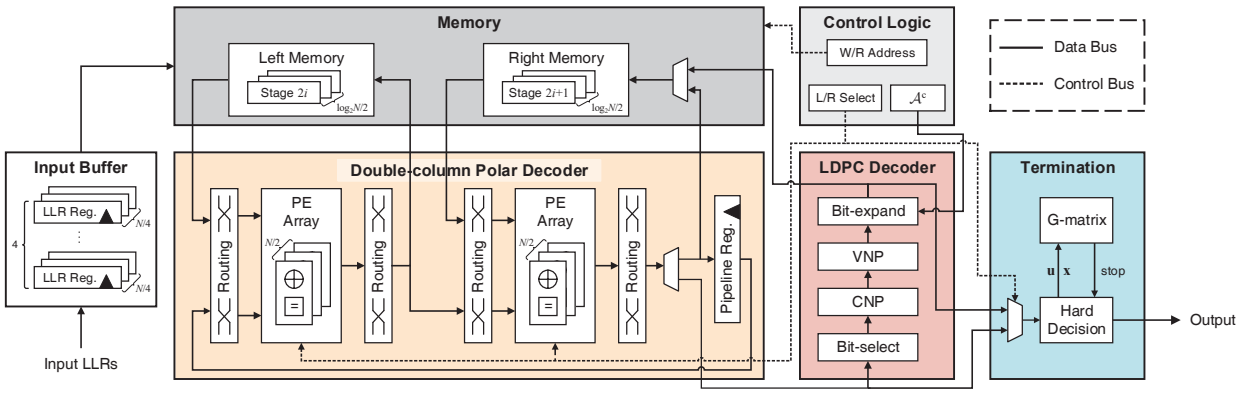


Fig. 10. Overall diagram of the proposed scalable polar OMS decoder.

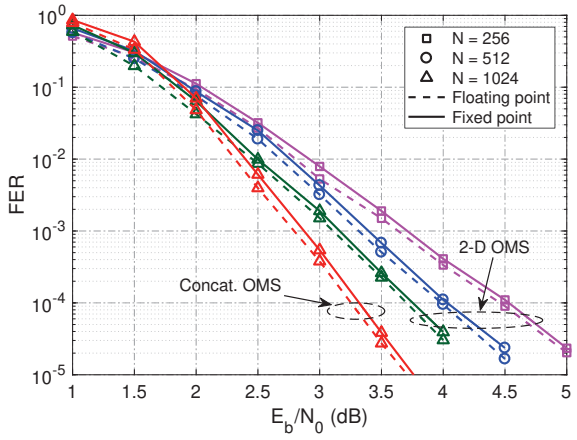


Fig. 11. Performance comparison for quantized 2-D OMS and concat. OMS decoders on various polar codes with $R = 1/2$.

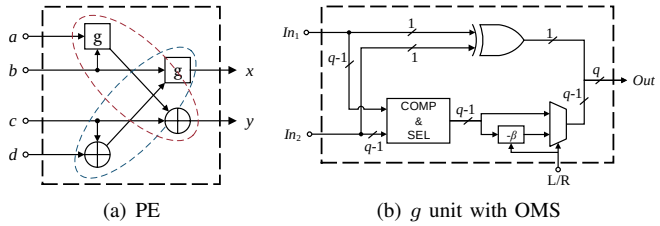


Fig. 12. Architecture of PE and offset-MS unit for polar decoder.

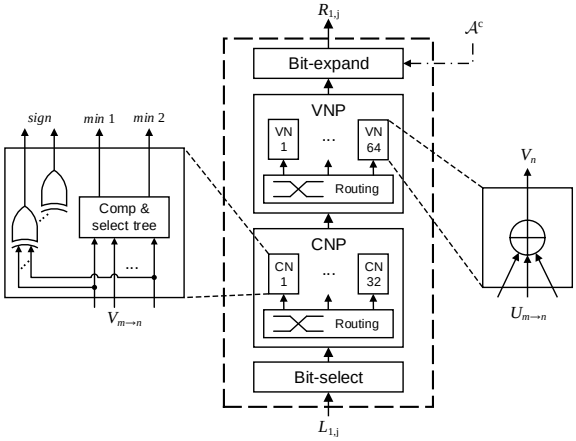


Fig. 13. Architectures of fully-parallel LDPC decoder.

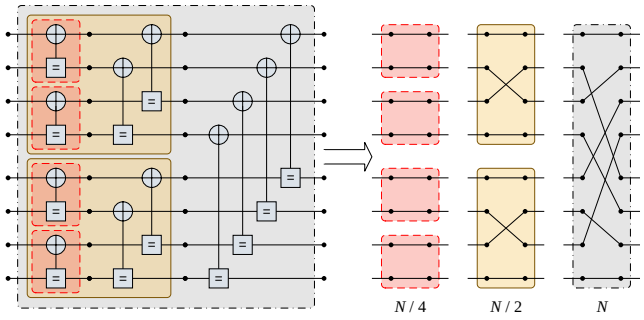
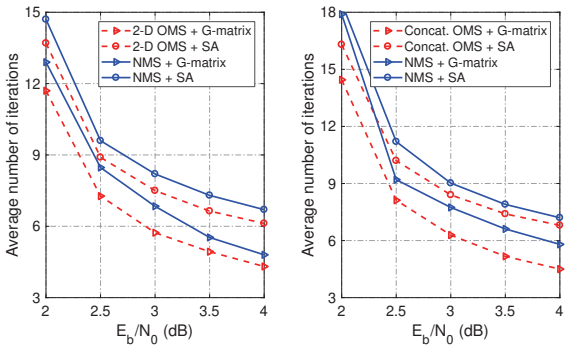


Fig. 14. Factor graph of polar code with $N = 8$ (left) and the routings of two consecutive stages for $N/4$, $N/2$, and N codes (right).

throughput of decoders, such as minLLR [5], G-matrix [5], and sign aided (SA) [6, 33]. Fig. 15 illustrates the average number of iterations for the G-matrix [5] and SA [6, 33] termination, where the SA scheme terminates the decoding process when the hard decision results of consecutive 3 iterations are identical. Compared to NMS, 2-D OMS and concat. OMS save about 10% to 20% iterations under the same termination scheme. Moreover, the G-matrix scheme saves about 30% iterations than SA. Therefore, G-matrix termination is adopted. The termination unit first makes the hard decision based on the most significant bit of Λ_j^u or Λ_j^x and encode the bits $\hat{\mathbf{u}}$ in the G-matrix unit. Then the stop signal is generated if the equality of $\hat{\mathbf{x}}$ and $\hat{\mathbf{u}}\mathbf{G}$ is satisfied or the maximum number of iterations is reached. A fully parallel polar encoder with $N \log N$ XOR gates is implemented in the G-matrix unit.

G. Timing Schedule

1) Decoding Schedule: The round-trip [26] scheduling is adopted to update the LLRs. During the iterative decoding, the PE array on the left side firstly processes shuffled L and R messages from the left memory and the pipeline register. Then the shuffled output of left PE array and shuffled LLRs from the right memory are processed by the right-side PE array. The results of the right PE array are written back to the right memory and the pipeline register. For the double-column polar



(a) Plain BP (b) Concatenated

Fig. 15. Average iterations of NMS, 2-D OMS, and concat. OMS decodings for $\mathcal{P}(1024, 512)$ and $\mathcal{P}_{\text{concat}}(1024, 512, 64, 32)$.

decoder [6, 33], the right-to-left propagation and the left-to-right propagation require $\log N/2$ cycles, respectively. Since one-stage pipelining is inserted into the memory, one extra cycle is needed for the data setup of polar decoder after the two propagations.

For concat. OMS decoder, the BP iteration of LDPC decoder between right-to-left propagation and left-to-right propagation can be overlapped with the data setup cycle after right-to-left propagation. Besides, the early termination check does not increase the overall clock cycles since the hard decision, and the termination check can be carried out simultaneously with the decoding process. The average decoding cycles of proposed polar OMS decoder are given by:

$$T_{\text{dec.}}(N, I_{\text{avg}}) = I_{\text{avg}} \times 2 \times (\lceil n/2 \rceil + 1), \tag{18}$$

where $n = \log N$ and I_{avg} is the average number of iterations.

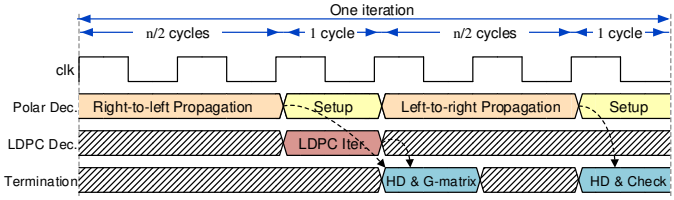


Fig. 16. Timing diagram of concatenated OMS decoder ($n = \log N$).

2) Pipelined Schedule: When the scalable polar OMS decoder is switched to 2-D OMS decoding mode for polar codes with length $N/4$ or $N/2$, 4 or 2 frames data are loaded into the decoder at each time. These frames have variable iteration numbers when passing the early termination check. The overall throughput is limited by the longest decoding latency among these frames since new data will be loaded only after all current frames are decoded. The pipelined schedule in Fig. 17 is presented to improve the throughput and hardware utilization. As illustrated in the figure, the frames that pass termination check are immediately sent to the output buffer, and the corresponding new LLR frames will be loaded from input buffer to the decoder in the next iteration. Meanwhile, the new incoming LLRs are loaded to the input buffer. The pipelined schedule ensures the decoder processes the LLRs with no additional stall cycles thus improves both the throughput and hardware utilization.

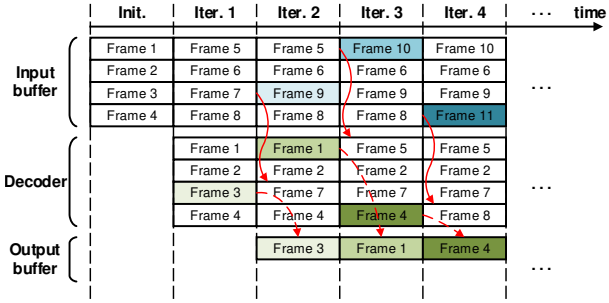


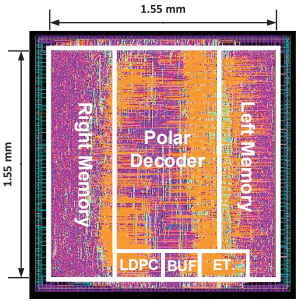
Fig. 17. Pipelined schedule for polar code $N = 256$.

VI. ASIC Results and Comparison

The ASIC implementation results and analysis of proposed decoders are presented in this section. The comparison with various SOA works is also given.

A. Implementation Details

The proposed scalable polar OMS decoder is implemented on 65 nm CMOS process and synthesized by Synopsys Design Compiler. The design is placed and routed using Synopsys IC Compiler. Random test vectors are generated to measure the switching activity. The power dissipation is estimated by Synopsys Prime Time PX. The maximum number of iterations is $T_{\max} = 20$ for 2-D OMS mode and $T_{\max} = 30$ for concat. OMS mode. Quantization schemes in Section V-B are used.



Technology	65 nm 1P9M CMOS
Quantization (bits)	7
Core Area (mm^2)	1.55×1.55
Gate Count	1332k
Supply Voltage (V)	1.1
Power (mW)	637
Clock Freq. (MHz)	270

Fig. 18. Layout and implementation results of scalable polar OMS decoder.

The layout and the implementation results of proposed scalable polar OMS decoder is shown in Fig. 18. A total of 1332k logic gates are integrated within 2.4 mm^2 core area. The decoder achieves a clock frequency of 270 MHz with a power dissipation of 637 mW and a supply voltage of 1.1 V.

The following metrics are used for fair comparison between different hardware architectures. Assuming the clock frequency is f_{clk}, and the average decoding cycles are T_{cycle}, the coded throughput of the given decoder is defined as:

$$\text{T/P (Gb/s)} = \frac{f_{\text{clk}} \times N}{T_{\text{cycle}}}. \tag{19}$$

The coded throughput with early termination scheme for different decoding modes of proposed decoder design is depicted in Fig. 19. The 2-D OMS decoding mode for $N = 256$ codes achieves the highest throughput of 8.5 Gb/s at 4 dB after using the pipelined schedule and early termination scheme.

TABLE I
HARDWARE COMPARISON FOR DIFFERENT DESIGNS OF POLAR BP DECODER

Decoder	This work					B. Yuan [5] [TSP, 2014]	Y. S. Park [6] [SVLSI, 2014]	S. M. Abbas [7] [TVLSI, 2016]	Y.-Y Chen [8] [TCAS-I, 2019]	K. Han [9] [TSP, 2019]
Process [nm]	65					45	65	45	40	65
Supply Voltage [V]	1.1					1.1	1.0	1.0	0.9	-
Architecture	Double-column Unidirectional			Double-column Concatenated		Iteration-level Overlapping	Double-column Unidirectional	Four-column Quarter-way	Double-column Bidirectional	Bit-wise Iterative
Flexibility	**Flexible**					Fixed length				
Code Length	$(256, 128)$	$(1024, 512)$								$(256, 128)$
Algorithm	2-D OMS			Concat. OMS		NMS	MS	NMS	NMS	Stochastic BP
E_b/N_0 @ FER $= 10^{-4}$	~ 4.6	$\sim$ **3.7**		$\sim$ **3.25**		~ 4.0	~ 3.9	~ 3.9	~ 3.9	~ 4.6
Quantization [bits]	7			$7, 6$		7	5	7	-	6
Avg. Iters.	3.7*	4.82*	4.25**	5.16*	4.50**	23.0*	6.57**	7.57*	7.48**	-
Frequency [MHz]	270					500	300	515	500	700
Gate Count† [kGE]	1332					1961	1025	2406	1053	221
Power [mW]	637					-	477.5	-	422.7	-
Decoding Cycles	37.0*	57.8*	51.0**	61.9*	54.0**	56.0*	65.7**	37.8*	67.32**	141*
T/P [Gb/s]	7.5	4.8	5.4	4.5	5.1	9.1	4.7	13.9	7.6	1.27
Scaled‡ to 65 nm and 1.0 V										
Decoding Latency [ns]	137.0	214.2	188.9	229.3	200.0	161.8	219.0	**106.0**	218.8	201.4
Scaled T/P [Gb/s]	7.5	4.8	5.4	4.5	5.1	6.3	4.7	**9.6**	4.7	0.64
Energy Eff. [pJ/b]	70.4	110.1	97.1	141.7	117.9	262.6	102.1	69.2	111.5	-
Hard. Eff. [Mb/s/kGE]	5.6	3.59	4.07	3.35	3.84	3.22	4.56	4.01	4.4	5.7

* Results reported at $E_b/N_0 = 3.5$ dB. ** Results reported at $E_b/N_0 = 4$ dB. † Estimated by 2-input NAND gate.

‡ Frequency $\propto S$ and energy $\propto \frac{1}{S}(\frac{1.0}{V_{dd}})^2$, where S is the scaling factor to 65 nm.

◇ Hardware efficiency = scaled throughput / gate count.

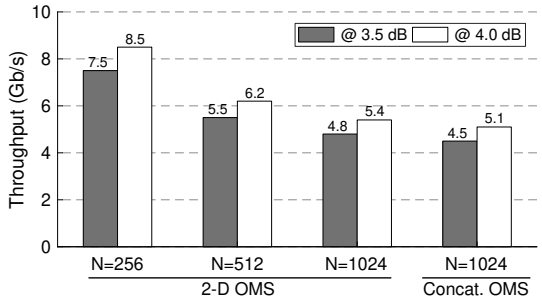


Fig. 19. Coded throughput of proposed scalable polar OMS decoder under different decoding modes.

B. Comparison with BP Decoders

Table I summarizes the comparison between the proposed scalable polar OMS decoder with SOA BP decoders [5]–[8] for $\mathcal{P}(1024, 512)$. The BP decoder based on stochastic computing [9] for $\mathcal{P}(256, 128)$ is also included. The results of [5], [7], and [9] are reported at $E_b/N_0 = 3.5$ dB while the results in [6] and [8] are reported at $E_b/N_0 = 4.0$ dB both with early termination. For the fair comparison, the results of the proposed 2-D OMS decoder and concatenated OMS decoder are calculated at both of these SNRs.

The proposed scalable polar OMS decoder has more flexibilities to decoder multi-length codes with two decoding algorithms. With 2-D OMS of code length $N = 256$, proposed decoder achieves a throughput of 7.5 GB/s with an average latency of 137 ns, outperforming the stochastic computing-based BP decoder in [9]. Under the 2-D OMS decoding mode for code length $N = 1024$, proposed scalable OMS decoder achieves comparable energy efficiency and hardware efficiency compared to other designs [5]–[8]. Compared to other polar BP decoders, 0.2 to 0.3 dB FER gain is observed on the proposed design. The gain comes from the DL optimizations which improve the error-correction performance of 2-D OMS decodings. Moreover, the error-correction performance near CA-SCL with list size 2 can be provided by switching to the concat. OMS mode for $N = 1024$.

TABLE II
HARDWARE COMPARISON WITH SCL AND CA-SCL DECODERS
($N = 1024$, $R = 1/2$)

Decoder	This work	[10]†	[11]‡	[12]‡	[13]‡
List size	-	$L = 2$			
Process [nm]	65	90	90	65	65
Gate Count [kGE]	1332	312	701	728	354
Frequency [MHz]	270	847	423	885	650
Decoding Cycles	**54.0***	2592.0	337.0	486.7	338*
Scaled to 65 nm					
Decoding Latency [ns]	**200***	2210	575	550	520*
Scaled T/P [Gb/s]	**5.1**	0.46	1.8	1.9	1.9
Hard. Eff. [Mb/s/kGE]	3.84	1.5	2.5	2.5	**5.5**

* Result reported at $E_b/N_0 = 4$ dB.
† Decoder without CRC. ‡ Decoders with CRC.

C. Comparison with SCL and CA-SCL Decoders

The comparison with the SOA SCL [10] and CA-SCL [11]–[13] decoders with $L = 2$ is in Table II. Due to the high parallelism of BP decoders, the proposed decoder reduces the decoding latency by 62% and improves the throughput by 168% compared to [13]. The resulting hardware efficiency of our design is 53% to 156% higher than those in [10]–[12].

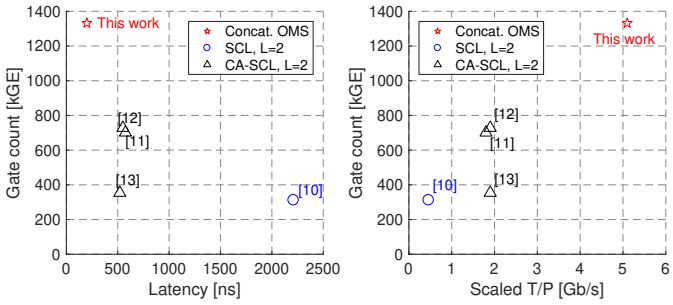


Fig. 20. Implementation comparison with SOA polar decoders.

Fig. 20 shows the relationship between the gate count and the latency as well as the gate count and the scaled throughput for different designs. The proposed scalable OMS decoder with concat. OMS mode has the lowest latency and highest scaled throughput compared to SOA decoders [10]–[13].

VII. Conclusion

In this work, we propose the DL methods to optimize the modified BP decodings, *e.g.* 2-D OMS and concatenated OMS. Given arbitrary code lengths, the DL methods, namely back-propagation and gradient descent, can effectively search the key parameters that result in good error-correction performance with acceptable complexity. Moreover, the scalable polar OMS decoder that supports both multi-length codes and the two decoding algorithms are implemented with 65 nm CMOS technology. Under 2-D OMS decoding mode, the decoder yields a throughput of 7.5 Gb/s on $N = 256$ codes and 5.4 Gb/s on $N = 1024$, which is comparable to the SOA polar BP decoders. Moreover, the concatenated OMS decoding mode with error-correction performance close to CA-SCL decoder of list size 2, achieves 5.1 Gb/s throughput and reduces the decoding latency to 200 ns.

References

[1] W. Xu, Z. Wu, Y.-L. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," in *IEEE Int. Workshop Signal Process. Syst. (SiPS)*, 2017, pp. 1–6.

[2] E. Arıkan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, 2009.

[3] 3GPP, "Final Report of 3GPP TSG RAN WG1 #87," Feb. 2017. [Online]. Available: http://www.3gpp.org/DynaReport/TDocExMtg{-}{-}R1-87{-}{-}31665.htm

[4] A. Pamuk, "An FPGA implementation architecture for decoding of polar codes," in *Int. Symp. Wireless Commun. Syst. (ISWCS)*, 2011, pp. 437–441.

[5] B. Yuan and K. K. Parhi, "Early stopping criteria for energy-efficient low-latency belief-propagation polar code decoders," *IEEE Trans. Signal Process.*, vol. 62, no. 24, pp. 6496–6506, 2014.

[6] Y. S. Park, Y. Tao, S. Sun, and Z. Zhang, "A 4.68 Gb/s belief propagation polar decoder with bit-splitting register file," in *Symp. VLSI Circuits Dig. Tech. Papers*, 2014, pp. 1–2.

[7] S. M. Abbas, Y. Fan, J. Chen, and C.-Y. Tsui, "High-throughput and energy-efficient belief propagation polar code decoder," *IEEE Trans. VLSI Syst.*, vol. 25, no. 3, pp. 1098–1111, 2017.

[8] Y.-T. Chen, W.-C. Sun, C.-C. Cheng, T.-L. Tsai, Y.-L. Ueng, and C.-H. Yang, "An integrated message-passing detector and decoder for polar-coded massive mu-mimo systems," *IEEE Trans. Circuits Syst. I Regul. Pap.*, vol. 66, no. 3, pp. 1205–1218, 2019.

[9] K. Han, J. Wang, W. J. Gross, and J. Hu, "Stochastic bit-wise iterative decoding of polar codes," *IEEE Trans. Signal Process.*, vol. 67, no. 5, pp. 1138–1151, 2019.

[10] A. Balatsoukas-Stimming, M. B. Parizi, and A. Burg, "LLR-based successive cancellation list decoding of polar codes," *IEEE Trans. Signal Process.*, vol. 63, no. 19, pp. 5165–5179, 2015.

[11] J. Lin, C. Xiong, and Z. Yan, "A high throughput list decoder architecture for polar codes." *IEEE Trans. VLSI Syst.*, vol. 24, no. 6, pp. 2378–2391, 2016.

[12] S. A. Hashemi, C. Condo, and W. J. Gross, "Fast and flexible successive-cancellation list decoders for polar codes," *IEEE Trans. Signal Process.*, vol. 65, no. 21, pp. 5756–5769, 2017.

[13] D. Kim and I.-C. Park, "A fast successive cancellation list decoder for polar codes with an early stopping criterion," *IEEE Trans. Signal Process.*, vol. 66, no. 18, pp. 4971–4979, 2018.

[14] E. Arıkan, "A performance comparison of polar codes and Reed-Muller codes," *IEEE Commun. Lett.*, vol. 12, no. 6, 2008.

[15] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, 2015.

[16] B. Yuan and K. K. Parhi, "Architecture optimizations for BP polar decoders," in *IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, 2013, pp. 2654–2658.

[17] J. Chen, A. Dholakia, E. Eleftheriou, M. P. Fossorier, and X.-Y. Hu, "Reduced-complexity decoding of LDPC codes," *IEEE Trans. Commun.*, vol. 53, no. 8, pp. 1288–1299, 2005.

[18] J. Zhang, M. Fossorier, and D. Gu, "Two-dimensional correction for min-sum decoding of irregular LDPC codes," *IEEE Commun. Lett.*, vol. 10, no. 3, pp. 180–182, 2006.

[19] A. Eslami and H. Pishro-Nik, "On finite-length performance of polar codes: stopping sets, error floor, and concatenated design," *IEEE Trans. Commun.*, vol. 61, no. 3, pp. 919–929, 2013.

[20] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 1, pp. 119–131, 2018.

[21] J. Guo, M. Qin, A. G. i Fabregas, and P. H. Siegel, "Enhanced belief propagation decoding of polar codes through concatenation," in *IEEE Int. Symp. Inf. Theory (ISIT)*, 2014, pp. 2987–2991.

[22] S. M. Abbas, Y. Fan, J. Chen, and C.-Y. Tsui, "Concatenated LDPC-polar codes decoding through belief propagation," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2017, pp. 1–4.

[23] Q.-p. Yu, Z.-p. Shi, L. Deng, and X. Li, "An improved belief propagation decoding of concatenated polar codes with bit mapping," *IEEE Commun. Lett.*, 2018.

[24] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, 2012.

[25] Y.-C. Liao, C.-C. Lin, H.-C. Chang, and C.-W. Liu, "Self-compensation technique for simplified belief-propagation algorithm," *IEEE Trans. Signal Process.*, vol. 55, no. 6, pp. 3061–3072, 2007.

[26] J. Xu, T. Che, and G. Choi, "XJ-BP: Express journey belief propagation decoding for polar codes," in *IEEE Global Commun. Conf. (GLOBECOM)*, 2015, pp. 1–6.

[27] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, p. 533, 1986.

[28] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[29] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, "Tensorflow: a system for large-scale machine learning," in *OSDI*, vol. 16, 2016, pp. 265–283.

[30] M. Benammar and P. Piantanida, "Optimal training channel statistics for neural-based decoders," in *IEEE Asilomar Conf. Signal, Syst. Comput.*, 2018, pp. 2157–2161.

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

[32] X.-Y. Hu, E. Eleftheriou, and D.-M. Arnold, "Regular and irregular progressive edge-growth tanner graphs," *IEEE Trans. Inf. Theory*, vol. 51, no. 1, pp. 386–398, 2005.

[33] S. Sun and Z. Zhang, "Architecture and optimization of high-throughput belief propagation decoding of polar codes," in *IEEE Int. Symp. Circuits Syst. (ISCAS)*, 2016, pp. 165–168.