NOLD: A Neural-Network Optimized Low-Resolution Decoder for LDPC Codes

Lei Chu, Huanyu He, Ling Pei, Robert C. Qiu

Abstract: **The min-sum (MS) algorithm can decode Low-density parity-check (LDPC) codes with low computational complexity at the cost of slight performance loss. It is an effective way to realize hardware implementation of the min-sum decoder by quantizing the floating belief messages (i.e., check-to-variable messages and variable-to-check messages) into low-resolution (i.e., 2-4 bits) versions. However, such a way can lead to severe performance degradation due to the finite precision effect. In this paper, we propose a neural-network optimized low-resolution decoding (NOLD) algorithm for LDPC codes to deal with the problem. Specifically, the optimization of decoding parameters (i.e., scaling factors and quantization step) is achieved in a hybrid way, in which we concatenate a NOLD decoder with a customized neural network. All learnable parameters associated with the decoding parameters are assigned to each neuron in the proposed method. What's more, we design a new activation function whose outputs are close to the employed quantizer ones when network parameters are finally optimized off-line. Finally, the performance of the proposed method is verified by numerous experiments. For the case of 2-bit decoding, the proposed approach significantly outperforms several conventional decoders at the expense of slightly increased off-line training time. Besides, the proposed method with 4-bit quantization incurs only 0.1dB performance loss compared with the floating min-sum decoder at the coded bit-error-rate of 10^{-5}. Moreover, we show that the proposed NOLD decoder works over a wide range of channel conditions for regular and irregular LDPC codes. Simulation code for reproductive results is publicly available[1].**

Index Terms: **Low-density parity-check codes, min-sum algorithm, low-resolution, neural network, optimization.**

I. INTRODUCTION

Low-density parity-check (LDPC) codes which were first discovered by Gallager [1] in the 1960s have been theoretically proved to achieve the performance near the Shannon-limit under belief propagation (BP) decoding algorithms [2]. It is believed that LDPC codes are of sustaining potential in the future wireless communication system, which is conceived to be ultra-fast, low-latency, and ultra-reliable [3]–[6].

The min-sum (MS) decoding algorithm has been widely studied as one of the BP decoding algorithms [7]. Compared with its original form, known as the sum-product (SP) decoding algorithm, the MS decoding algorithm has a much lower computational complexity at the cost of slight performance degradation. The performance loss can be compensated by using a normalization factor or an offset factor [8], [9]. These two modified versions of the MS decoding algorithm are called normalized min-sum (NMS) decoding algorithm [8] and offset min-sum (OMS) decoding algorithm [9], respectively. The scaling min-sum (SMS) decoding algorithm uses normalization factor and offset factor in synergy and can achieve better performance. Besides, they can narrow the oversized check-to-variable messages to approximate the SP algorithm.

The effects of scaling factors have been analyzed in [10]. It has been shown that SMS outperforms MS by approximating the check-to-variable messages to the SP algorithm. For the regular LDPC codes, fixed scaling factors can be used throughout the decoding process for each node since the performance is insensitive to the scaling factors. The work of [11] proposed a numerical method that directly computes the theoretical values of fixed scaling factors. However, fixed scaling factors are no longer effective for the irregular case. In general, it is supposed to vary the scaling factors from one node to another as well as from one iteration to the next [12].

The typical way is to optimize the values of scaling factors by the adaptive strategies, which design an adaptive rule that computes the optimal scaling factors from the decoder's feedback at each iteration. The work of [13] utilized a pre-configured look-up table, wherein wherein the proportion of unsatisfied parity-check equations adaptively determines the scaling factors. It has been shown in [14] that the scaling factors increase exponentially with the iterations and the final values are equal to 1. As a result, the authors of [14] used an exponential equation to calculate the scaling factors. In [15], The adaptive scaling factors are computed by the ratio of the second minimum and the maximum values among the variable-to-check messages. It is noted that these adaptive methods cannot effectively jointly optimize decoding parameters. Moreover, such adaptive processes add some complexity and increase the decoding delay.

Alternatively, the optimal value of the scaling factors can be determined by a probabilistic analytical method called density evolution (DE). The works of [16], [17] employed this numerical method to estimate the optimal scaling factors. In [16], the DE method was used to obtain the optimal scaling factors and then estimated the achievable performance gain induced by the scaling factor. Authors of [17] specified the DE optimization of two-dimensional scaling factors for the scaling min-sum algorithm. Afterward, they introduced a method to determine the two-dimensional scaling factors. The work of [18], [19]

Manuscript received XX approved for publication by XX, Co-Editor-in-Chief, DATE.

Dr. Chu's work is supported by the International Postdoctoral Exchange Fellowship Program from the China Postdoctoral Council. The work of Dr. Qiu and Dr. Pei are partially supported in part by the N.S.F. of China under Grant 61571296 and 61873163, respectively.

Lei Chu and Huanyu He contribute equally to this work. The authors are with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China.

Lei Chu is the corresponding author.

Digital Object Identifier XX

[1]https://github.com/Leo-Chu/NOLD

analyzed the bit error rate (BER) performance of the LDPC codes under the Rayleigh fading channel and asymmetric memoryless channels by DE, respectively. Nevertheless, there exist two problems with the DE method. On the one hand, the basic premise of DE is that the analyzed LDPC code has infinite code length or cycle-free Tanner graph under the condition of the symmetry channel. On the other hand, searching for the optimal scaling factors based on the DE method becomes intractable when the number of different node degrees is large.

On the other hand, it is an effective way to realize hardware implementation of the min-sum decoder by quantizing the floating belief messages into low-resolution (i.e., 2-4 bits) versions. A very recent work [25] proposed to reduce the hardware complexity by adopting the novel compressed structure of the network. As shown in the work of [24], a dynamic quantization can achieve better performance than a fixed quantization. Well-designed adaptive rules can dynamically determine the quantization parameters. Authors of [26] proposed an adaptive quantization scheme called quasi-uniform quantization that can match the dynamic range of belief messages. The quasi-uniform quantization uses a uniform quantization interval for the message with a small magnitude and an exponentially increasing interval for the message with a large magnitude. In our previous work [28], the uniform quantization intervals are updated every iteration based on a low-complexity adaptive rule that calculates the numbers of satisfied check nodes. Another way to estimate the quantization parameters is the well-known discrete density evolution (DDE) method [29], which can evaluate the low-resolution performance of the BP decoder and design the quantizer for irregular LDPC codes in [30], [31].

Except for the adaptive optimization and DE optimization of decoding parameters, deep learning-based methods are of great potential since recent technological advancements make deep learning meet the requirements of future wireless networks [32], [33]. Authors of [34] utilized a machine learning technique to optimize the scaling factors of the scaling min-sum algorithm. The work of [35] established a neural network decoder with similar topological construction to the Tanner graph. The neurons corresponding to the edges in the Tanner graph are assigned in the trainable weights. In [38], the extrinsic messages of the BP decoder is jointly optimized by the supervised learning method and the annealing method. A novel decoding architecture proposed in [39], embedding the convolutional neural network (CNN) with a BP decoder, achieves significant performance gains under additive correlated Gaussian noise (ACGN). With the novel Finite-Alphabet Message Passing scheme [27], the authors proposed to optimize the classical iterative decoder of LDPC codes. Besides, the experimental results in [27] proved that the proposed method with 3-bit precision achieves comparable results with the min-sum algorithm.

The deep learning based decoding algorithms [6], [27], [33], [34], paving a new way to decoding LDPC codes, can achieve superior performance at the cost of slightly higher computational complexity when compared to classical ones. In this paper, we propose a new neural-network optimized low-resolution LDPC decoder. The contributions of this work can be summarized as follows.

I. To reduce the min-sum decoder's hardware cost, we propose to quantize the floating belief messages (i.e., check-to-variable messages and variable-to-check messages) into low-resolution (i.e., 2-4 bits) versions which are beneficial for hardware implementation. To deal with the finite precision effect, we propose a new neural network-based method. The decoding parameters, including scaling factors and quantization step for variable node and check node output messages, are jointly optimized.

II. To fully leverage the advantages of deep learning-based methods, we propose a new activation function whose outputs are close to the employed quantizer ones when network parameters are finally optimized offline. Besides, with the new activation function, the current decoder can be extended to an arbitrary resolution.

III. We verify the performance of the proposed method with numerous case studies. Our results reveal that the proposed decoder enables low-resolution belief messages without a significant performance loss in terms of coded bit error rate performance. What's more, it is shown that, for both regular and irregular LDPC codes, the proposed approach significantly outperforms several conventional decoders over a wide range of channel conditions.

For reproductive results and possible extensions of the proposed method by interested readers, the decoder developed in this paper has been made publicly available[2].

The rest of the paper is organized as follows. Section II describes the system model. In Section III, we first present the MS decoding and then introduce the modification of MS decoding that operates scaling and quantization on the belief message of each node, which serves as the fundamental decoding scheme of our proposed algorithm. Section IV and Section V elaborates the proposed NOLD decoder and present the simulation results, respectively. Finally, Section VI concludes this paper.

II. System Model

In this part, we introduce the system model of the low-resolution LDPC decoding. We consider the BPSK modulation over the three noise models.

A. LDPC Codes

At the transmitter, the base-band signals $\mathbf{s}$ of length K are encoded to binary coded bits $\mathbf{u}$ of length N. The encoded bits $\mathbf{u}$ is represented by

$$\mathbf{u} \in \left\{ \mathbf{c} \in \mathbb{F}_2^{\mathbb{N}} \left| \mathbf{H}\mathbf{c}^{\mathbf{T}} = \mathbf{0} \right. \right\}, \tag{1}$$

where $\mathbf{H} \in \mathbb{F}_2^{\mathbb{M}\times\mathbb{N}}$ is a binary parity check matrix with M rows, N columns and E 1-elements. The associated Tanner graph $\mathbf{G}$ define by $\mathbf{H}$ has N variable nodes, M check nodes and E edges. The variable node v is connected by an edge e to the check node c if the corresponding element in $\mathbf{H}$ is equal to 1. The number of check nodes (variable nodes, resp.) that are connected to a variable node v (check node c, resp.) is termed the degree of variable node v (check node c, resp.) and is different from one node to another in an irregular LDPC code. Since $\mathbf{H}$ is a sparse matrix, the number of the edges E is far less than $M \times N$.

[2]https://github.com/Leo-Chu/NOLD

B. Channel Noise Models

The N coded bits $\mathbf{u}$ are modulated by a binary-to-bipolar function where the binary signals $\{0,1\}^N$ are converted to $\{1,-1\}^N$. Then the BPSK modulated signals $\mathbf{x}$ are sent through an additive white Gaussian noise (AWGN) channel:

$$\mathbf{y} = \mathbf{x} + \mathbf{n}, \tag{2}$$

where the noise vector $\mathbf{n}$ follows Gaussian distribution with zero mean and σ^2 variance.

However, practical channels can include fading noise and correlated noise caused by multi-path propagation and inter-symbol interference. The LDPC codes will confront severe performance degradation if the decoder can not deal with the fading and correlated noise.

B.1 Rayleigh Fading Noise Channel

The magnitude of transmit signal will vary randomly or fade when passing through a Rayleigh fading channel. Commonly, a Rayleigh fading model assumes that the real and imaginary parts of the received signal follow zero-mean Gaussian distribution so that the sum of such two independent and identically Gaussian random variables follows a Rayleigh distribution [18], [20]. The Rayleigh fading channel outputs are defined by:

$$\mathbf{y} = \mathbf{r}\mathbf{x} + \mathbf{n}, \tag{3}$$

where $\mathbf{r}$ is known to the receiver as the Rayleigh fading factor vector whose entries have density probability shown as $p(r) = 2r \cdot e^{-r^2}$.

B.2 Additive Correlated Gaussian Noise Channel

We analyze one-sided ACGN that has correlation along the temporal dimension instead of the block dimension. The correlated noise is modeled by $\hat{\mathbf{n}} = \Phi^{\frac{1}{2}}\mathbf{n}$ where $\mathbf{n}$ is an noise vector of length N and Φ is an $N \times N$ matrix. The entries of $\mathbf{n}$ are i.i.d. Gaussian variables and are turned to correlated Gaussian variables by multiplying correlation matrix $\Phi^{\frac{1}{2}}$ [23].

To keep the same variance of $\hat{\mathbf{n}}$ as $\mathbf{n}$, we consider $\frac{1}{K}\mathrm{tr}(\Phi) = 1$. Besides, we use an exponential covariance model to define the entries of Φ:

$$\Phi_{i,j} = \begin{cases} \varphi^{j-i}, & i \le j \\ \left(\varphi^{i-j}\right)^*, & i \ge j \end{cases}, \tag{4}$$

where $\Phi_{i,j}$ represents the (i,j)th entry of Φ, and φ is the correlation factor with $|\varphi| < 1$.

The ACGN channel outputs are presented as:

$$\mathbf{y} = \mathbf{x} + \Phi^{\frac{1}{2}}\mathbf{n}. \tag{5}$$

Finally, the channel outputs $\mathbf{y}$ are sent to the decoder from the receiver.

C. Quantized Decoding

A quantizer is required for the LDPC decoder in consideration of the hardware implementation. The quantizer can output belief messages that take finite discrete values and reduce the power consumption and storage cost [21], [22]. Commonly, $4 \le b \le 7$ are suitable values (see, e.g. [30] and references therein). The b-bit quantization is shown as follow:

$$Q(x) = \begin{cases} q_1 & x \in T_1 : (-\infty, t_1] \\ \vdots & \vdots \\ q_k & x \in T_k : (t_{k-1}, t_k] \\ q_{k+1} & x \in T_{k+1} : (t_k, t_{k+1}] \\ \vdots & \vdots \\ q_{2^b} & x \in T_{2^{b-1}} : (t_{2^b-1}, \infty] \end{cases}. \tag{6}$$

For simplicity, we shall model the quantizer as symmetric uniform quantizer with step size Δ and quantization bit b. We start by defining a set of quantization labels $Q = \{q_1, q_2, \cdots, q_{2^b}\}$ with entries

$$q_k = \frac{\left(2k - 2^b - 1\right)\Delta}{2}, k = 1, 2, \cdots, 2^b. \tag{7}$$

Moreover, let $T = \{-\infty, t_1, \cdots, t_{2^b-1}, \infty\}$ specify the set of quantization thresholds. For uniform quantizers, the quantization thresholds are given by

$$t_k = \frac{\left(2k - 2^b\right)\Delta}{2}, k = 1, 2, \cdots, 2^b - 1. \tag{8}$$

The uniform quantization can be uniquely determined by the set of quantization labels Q and the set of quantization thresholds T. Therefore, the mapping function $Qu(\cdot)$ can also be described by the quantization step size Δ and quantization bit b. The quantizer maps the floating inputs into the quantized outputs in the following way:

$$Qu(x) = \mathrm{sgn}(x) \cdot \Delta\left(\left\lfloor \frac{|x|}{\Delta} \right\rfloor + \frac{1}{2}\right), \tag{9}$$

where $\lfloor \cdot \rfloor$ is the rounding down function. The messages that have values smaller than q_1 or larger than q_{2^b} are respectively saturated to q_1 and q_{2^b}.

Although the finite precision effect exacerbates the decoding performance, the uniform quantizer has low hardware complexity and is simple to be implemented. Moreover, the decoding system with uniform quantization obviates the floating-point operations and lookup table storing , reducing the hardware complexity and store memory.

III. Min-Sum Decoding Scheme

This section introduces the decoding scheme of our work. To elucidate our proposed algorithm, we first introduce the MS decoding. Then, we will describe MS decoding with low-resolution belief messages.

A. Min-Sum Decoding

At the start of the MS decoding, the log-likelihood ratios(LLRs) computed from the channel outputs are sent to the variable nodes as initial variable-to-check messages for the first decoding iteration [7]. The LLRs of MS decoder l_v^0 are the channel outputs y_v for $v = 1, 2, \cdots, N$:

$$l_v^0 = y_v. \tag{10}$$

At the following iteration for $i = 1, 2, \cdots, I$ where I represents the maximum number of iterations, the check-node messages and variable-node message are updated as follow:

$$l_{c,v}^{i} = \prod_{v' \in \mathcal{N}(\rfloor) \backslash \sqsubseteq} \operatorname{sgn}\left(l_{v',c}^{i-1}\right) \cdot \min_{v' \in \mathcal{N}(\rfloor) \backslash \sqsubseteq} \left|l_{v',c}^{i-1}\right|, \tag{11}$$

$$l_{v,c}^{i} = l_{v}^{0} + \prod_{c' \in \mathcal{M}(v) \backslash c} l_{c',v}^{i}, \tag{12}$$

where $l_{c,v}^{i}$ ($l_{v,c}^{i}$, resp.) represents the message passed from check node c (variable node v) to variable node v (check node c) at iteration i. $\mathcal{N}(\rfloor)$ ($\mathcal{M}(\sqsubseteq)$, resp.) denotes the set of neighboring variable nodes (check nodes, resp.) of the check node c (variable node v, resp.).

When the Ith iteration arrives, the check-to-variable messages update as (11) but the variable-to-check updating is a bit different from (12). The variable node combines all the check-to-variable messages and computes the final decoding outputs $\hat{x}_v$ through a hard decision. The estimated code bits $\hat{u}_v$ can be obtained after demodulation:

$$l_{v}^{i} = l_{v}^{0} + \prod_{c' \in \mathcal{M}(\sqsubseteq)} l_{c',v}^{i}, \tag{13}$$

$$\hat{x}_v = \operatorname{sgn}\left(l_{v}^{i}\right), \tag{14}$$

$$\hat{u}_v = \mathcal{D}\left(\hat{\S}_{\sqsubseteq}\right), \tag{15}$$

where $\operatorname{sgn}(\cdot)$ represents the sign function and $\mathcal{D}(\cdot)$ is a BPSK demodulator that converts bipolar signals $\{1, -1\}^N$ to binary signals $\{0, 1\}^N$.

B. Modification of Min-Sum Decoding

Though the MS decoding simplifies the check-to-variable processing, it overestimates the check-to-variable messages and causes a performance loss compared with the SP decoding. The scaling MS algorithm compensates for the performance loss by narrowing the check-to-variable messages. This modified MS algorithm processes the belief messages in a one-dimensional manner without considering the variable-to-check messages. However, the outgoing messages from variable nodes are also the incoming messages to the check nodes in the next iteration. Thus, the variable-to-check messages also play an essential role in the iterative process, which suggests processing both the check-to-variable messages and variable-to-check messages [17]. This two-dimensional scaling MS algorithm can outperform the one-dimensional scaling MS algorithm. It is rational to apply quantization to the check node and variable node in the same light when considering the low-resolution case.

Let $Q_x(\cdot)$ ($Q_y(\cdot)$, resp.) be the quantizer for check-to-variable messages (variable-to-check messages, resp.). α_x and β_x (α_y and β_y, resp.) denote the normalization factor and offset factor for check-node processing(variable-node processing, resp.), respectively. i represents the iteration numbers for $i = 1, 2, \cdots, I$. The channel initialization is shown as:

$$l_{v}^{0} = Q_{y}^{0}\left[\max\left(\alpha_{y}^{0} \cdot y_v + \beta_{y}^{0}, 0\right)\right]. \tag{16}$$

The check-node processing and variable-node processing are described as:

$$l_{c,v}^{i} = Q_{x}^{i}\left[\left(\prod_{v' \in \mathcal{N}(c)/v} \operatorname{sgn}\left|l_{v',c}^{i-1}\right|\right) \cdot \max\left(0, \alpha_{x}^{i} \cdot \min_{v' \in \mathcal{N}(c)/v}\left|l_{v',c}^{i-1}\right| + \beta_{x}^{i}\right)\right], \tag{17}$$

$$l_{v,c}^{i} = Q_{y}^{i}\left[l_{v}^{0} + \max\left(0, \alpha_{y}^{i} \cdot \sum_{c' \in \mathcal{M}(v)/c} l_{c',v}^{i} + \beta_{y}^{i}\right)\right]. \tag{18}$$

When it reaches the maximum iteration number I, the variable-to-check processing at iteration I can be represented by (13). As shown in (17) and (18), within one iteration, the outgoing messages from check nodes or variable nodes are scaled and quantized equally for each node. However, it is not efficient for irregular LDPC codes, and we should apply different scaling and quantization to each node for irregular LDPC codes. In this case, an effective method is required to optimize numerous decoding parameters.

IV. Proposed Decoding Algorithm

A. Proposed Neural Network

In [35], a deep neural network constructed by directly unfolding a Tanner graph optimizes the network parameters and processes the input signals just as an iterative BP decoder. In this part, we extend the trellis representation of the BP decoder to the proposed decoder. Similar to [35], we assign learnable parameters associated with the decoding parameters to each neuron in our neural network termed as NOLD network.

In the NOLD network, the dimension of the input layer and output layer is N, which is the number of the variable nodes (i.e., the code block length). The input layer receives the channel outputs, and the output layer outputs the final marginalization. The other layers are the hidden layers that contain E neurons associated with the edges in the Tanner graph. The NOLD network with $2I$ hidden layers graphically represents the NOLD decoder with I total iterations.

We define the outputs of the vth neuron corresponding to the variable node v in the input layer (output layer, resp.) as x_{v}^{In} (x_{v}^{Ot}, resp.). We use $x_{e_{v,c}}^{Hd_i}$ to denote the outputs of the neuron corresponding to the edge $e_{v,c}$ that connects the variable node v and the check node c in the ith hidden layer.

In the input layer, the neurons receive the channel ouputs y_v for $v = 1, 2, \cdots, N$:

$$x_{v}^{In} = y_v. \tag{19}$$

For the first hidden layer (i.e, $i = 1$), the neuron associated with the edge $e_{v,c}$ is connected to the input neuron corresponding to the variable node v. This layer outputs the quantized and scaled LLRs:

$$x_{e_{v,c}}^{Hd_1} = Q_{v}^{1}\left[\max\left(0, \alpha_{v}^{1} \cdot x_{v}^{In} + \beta_{v}^{1}\right)\right]. \tag{20}$$

For odd values of $i\,(i>1)$, the neuron associated with the edge $e_{v,c}$ is connected to all the neurons corresponding to the edges $e_{v,c'}$ for $c' \in \mathcal{M}\left(\sqsubseteq\right)\backslash\rfloor$ in the $i-1$th layer and is additionally connected to the neuron corresponding to the edge $e_{v,c}$ in the first hidden layer. The neurons in this layer deal with the messages in the same way as the variable-to-check processing:

$$x_{e_{v,c}}^{Hd_i}=Q_{e_{v,c}}^{i}\left[x_{e_{v,c}}^{Hd_1}+\max\left(0,\right.\right.$$
$$\left.\left.\alpha_{e_{v,c}}^{i}\cdot\sum_{c'\in\mathcal{M}(v)/c}x_{e_{v,c'}}^{Hd_{i-1}}+\beta_{e_{v,c}}^{i}\right)\right]. \quad (21)$$

For even values of i, the neuron associated with the edge $e_{v,c}$ is connected to all the neurons corresponding to the edges $e_{v',c}$ for $v' \in \mathcal{N}\left(\rfloor\right)\backslash\sqsubseteq$ in the $i-1$th layer. The outputs of this layer are equivalent to check-to-variable messages given by:

$$x_{e_{v,c}}^{Hd_i}=Q_{e_{v,c}}^{i}\left[\left(\prod_{v'\in\mathcal{N}(c)/v}\operatorname{sgn}\left|x_{e_{v',c}}^{Hd_{i-1}}\right|\right)\cdot\max\left(0,\right.\right.$$
$$\left.\left.\alpha_{e_{v,c}}^{i}\cdot\min_{v'\in\mathcal{N}(c)/v}\left|x_{e_{v',c}}^{Hd_{i-1}}\right|+\beta_{e_{v,c}}^{i}\right)\right]. \quad (22)$$

The neuron that is associated with the vth variable node in the output layer is connected to all the neurons related to the edges $e_{v,c}$ for $c \in \mathcal{M}\left(\sqsubseteq\right)$ in the last hidden layer and is also connected to the neuron corresponding to the edge $e_{v,c}$ in the first hidden layer. The final outputs of the network are shown as:

$$x_v^{Ot}=x_{e_{v,c}}^{Hd_1}+\sum_{c\in\mathcal{M}(v)}x_{e_{v,c}}^{Hd_{2I}}. \quad (23)$$

In this network, the trainable variables are scaling factors and quantization parameters. The trainable scaling factors are α_v^1, β_v^1 for $v=1,2,\cdots,N$ and $\alpha_{e_{v,c}}^{i}$, $\beta_{e_{v,c}}^{i}$ for $e_{v,c}=1,2,\cdots,E$, $i=2,3,\cdots,2I$. The trainable quantization parameters are quantization level sets $Q_{e_{v,c}}^{i}$ and in particular for the uniform quantization, are the quantization steps $\Delta_{e_{v,c}}^{i}$ for $e_{v,c}=1,2,\cdots,E$, $i=1,2,\cdots,2I$.

B. Neural Quantizer

The problem in the neural network is that the quantization function is non-differentiable at some points and its derived function is zero almost everywhere. This obstacle gets us in trouble for obtaining the gradient information from the backward computation process. Hence we can not evaluate the gradients by the backward propagation mechanism. A neural quantizer is needed to be designed for the neural network to train the learnable parameters.

To tackle this difficulty, we introduce a soft quantization function to the neural quantizer [36], [37]. The soft quantization function is suitable for the neural quantizer since it has non-zero derivative values and non-differentiable points. We assume a b-bit precision neural quantizer. The soft quantization function is represented by $Qn\left(\cdot\right)$. Given an input x, the quantized outputs

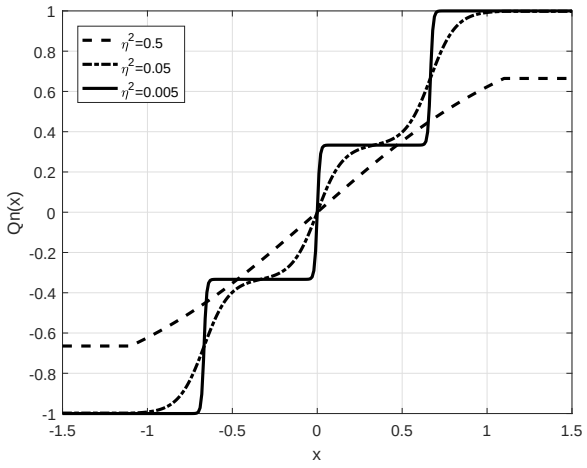


Fig. 1. The shapes of the soft quantization function under different η^2, $b=2$, $Q=\left\{-1,-\frac{1}{3},1,\frac{1}{3}\right\}$.

can be written as:

$$Qn\left(x\right)=\frac{\sum_{k=1}^{2^b}q_k\cdot e^{-\frac{(x-q_k)^2}{2\eta^2}}}{\sum_{k=1}^{2^b}e^{-\frac{(x-q_k)^2}{2\eta^2}}}, \quad (24)$$

where $Q=\{q_1,q_2,\cdots,q_{2^b}\}$ is the quantization level set of the b-bit neural quantizer and η is a shaping factor. It is also noteworthy that the soft quantization function (24) is the minimum mean square error (MMSE) estimator for a discrete signal over the AWGN channel [38].

A soft quantization function is determined by the quantization level set and smoothness factor. Thus, if the quantization levels are symmetric concerning the origin and each quantization level has the same intervals to its adjacent quantization levels, we can construct a uniform quantization. Fig.1 shows a two-bit uniform quantization constructed by the soft quantization function. We keep the quantization level set unchanged and tune the shaping factor. It is depicted that the shaping factor significantly affects the shape of the function. As shown in Fig.1, when $\eta^2\to\infty$, the soft quantization asymptotically approaches the linear function and when $\eta^2\to 0$, the soft quantization degenerates to a hard staircase quantization. Since the smoothness of the soft quantization function is controlled by the shaping factor η^2, the shaping factor η^2 should be tailored to the determined quantization level set Q. The shaping factor is set to a small number very close to zero according to the number of quantization levels at the training stage. It will be fixed to zero (i.e., hard staircase quantization) during the inference stage.

In our NOLD network mentioned above, each layer's activation function is replaced by the soft quantization function, increasing the neural network's generalization ability by introducing nonlinearity and can train the quantization parameters through the quantization activation function.

C. Loss Function

A proper loss function significantly affects the final results. In [35], the sigmoid function is used to preprocess the outputs

from the neural network before a cross-entropy function is used to calculate the loss. In [39], a normality test term is introduced to the loss function so that the distribution similarity between the residual noise and Gaussian noise can be evaluated. By minimizing the normality test, the distribution of a network's outputs is shaped to approximate Gaussian distribution.

For the final output layer, we map the final marginalization to the code bits by the activation function which replaces the hard decision in the standard BP algorithm. However, the sign function we used in the BP decoder has zero derivation almost everywhere except for some non-differentiable point. We can obtain a soft sign function by utilizing the soft quantization function in (24). We set $b = 2$, $Q = \{-1, 1\}$ and get a neural sign function $\mathrm{nsgn}(\cdot)$:

$$\mathrm{nsgn}(x) = \frac{e^{-\frac{(x+1)^2}{2\eta^2}} - e^{-\frac{(x-1)^2}{2\eta^2}}}{e^{-\frac{(x+1)^2}{2\eta^2}} + e^{-\frac{(x-1)^2}{2\eta^2}}} = \frac{e^{-\frac{2x}{\eta^2}} - 1}{e^{-\frac{2x}{\eta^2}} + 1}. \tag{25}$$

The BPSK demodulator $\mathcal{D}(\cdot)$ receives the neural decision and outputs the estimated code bits. We use $\theta(\cdot)$ to represent this final process:

$$\theta(x) = \mathcal{D}(\mathrm{nsgn}(\S)) = \frac{\infty - \mathrm{nsgn}(\S)}{\in} = \frac{\infty}{\infty + \rceil^{-\frac{\in \S}{\eta \in}}}. \tag{26}$$

Note that, if we set $\eta^2 = 2$, $\theta(\cdot)$ degenerates to a sigmoid function. The sigmoid activation function used in the output layer estimates the belief probability distribution of the code bit. Since the target outputs u_v and actual outputs o_v belong to interval $[0, 1]$, we use the cross-entropy function to calculate the loss:

$$Loss = -\frac{1}{N}\sum_{v=1}^{N}(u_v \log(o_v) + (1 - u_v)\log(1 - o_v)). \tag{27}$$

The cross-entropy loss function, which represents a measure of the divergence between u_v and o_v, has a fast convergence by using the gradient descent method [32]. The NOLD network minimizes the cross entropy in (27) and gets the optimized values of the decoding parameters.

D. The Proposed Neural-network Optimized Low-resolution Decoding Algorithm

The frame-chart of the proposed NOLD decoder is shown in Fig.2, which can be divided into two stages: Offline training and Online decoding. At the offline training stage, the proposed NOLD network tries to obtain a global optimization of the decoding parameters. The network contains a four-layer network that optimizes one-iteration parameters. The network receives the irrelevant messages as the training data from the concatenated NOLD decoder. The decoding parameters that best fit the training data are chosen to configure the NOLD decoder for the next iteration. The network optimization will not stop until some criteria are satisfied. For the online decoding stage, the trained parameters (scaling factors and the quantization parameters) are sent to the decoder for further inference. The detailed analysis will be given in the following.

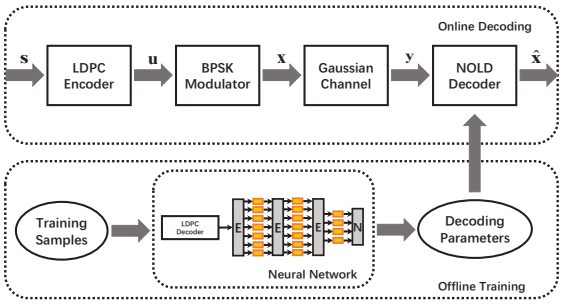


Fig. 2. The frame-chart of the proposed neural-network optimized low-resolution decoding algorithm.

We start by introducing some assumptions. Let an irregular LDPC code has variable-node degree distribution $\lambda(x) = \sum_{d=1}^{dv_{\max}} \lambda_d x^{d-1}$ and check node degree distribution $\rho(x) = \sum_{d=1}^{dc_{\max}} \rho_d x^{d-1}$. Let $dv(v)$ be the degree of the variable node v for $v = 1, 2, \cdots, N$ and $dc(c)$ be the degree of the check node c for $c = 1, 2, \cdots, M$. α_{dv}^i and β_{dv}^i denote the normalization factor and the offset factor for the variable nodes with degree dv at decoding iteration i for $dv = 1, 2, \cdots, dv_{\max}$. α_{dc}^i and β_{dc}^i denote the normalization factor and the offset factor for the check nodes with degree dc at decoding iteration i for $dc = 1, 2, \cdots, dc_{\max}$. $q_{dv}^i(\cdot)$ represents the ith-iteration quantization for variable nodes with degree dv for $dv = 1, 2, \cdots, dv_{\max}$. $q_{dc}^i(\cdot)$ represents the ith-iteration quantization for check nodes with degree dc for $dc = 1, 2, \cdots, dc_{\max}$.

The network has a four-layer architecture that represents one-iteration decoding. The first layer, called the input layer, has E neurons corresponding to the edges in the Tanner graph. Each neuron associated with the edge $e_{v,c}$ in the input layer receives the ith-iteration check-to-variable message that passes through the edge $e_{v,c}$ from the NOLD decoder:

$$x_{e_{v,c}}^{In} = l_{c,v}^i. \tag{28}$$

The second layer (i.e., the first hidden layer) carries out a similar procedure to the ith(i odd) hidden layer. Each neuron receives the extrinsic message in this layer, which is quantized and scaled LLR from the NOLD decoder. Besides, each neuron casts up all intrinsic messages from all connected neurons in the input layer. For variable-irregular LDPC codes, the variable-to-check messages should be scaled and quantized considering the degree differences among the adjacent variable nodes. Thus, we assign the same parameters to the neurons associated with the variable nodes that have the same degree:

$$x_{e_{v,c}}^{Hd_1} = Q_{dv(v)}^i\left[l_v^0 + \max\left(0, \alpha_{dv(v)}^i \cdot \sum_{c' \in \mathcal{M}(v)/c} x_{e_{v,c'}}^{In} + \beta_{dv(v)}^i\right)\right]. \tag{29}$$

In the third layer, that is, the second hidden layer, the connection of the neurons to the previous layer is similar to the

ith(i even) hidden layer. For check-irregular LDPC codes, we also consider the check-dimensional optimization of decoding parameters for different check node degrees. The outputs of this layer are:

$$x_{e_{v,c}}^{Hd_2} = Q_{dc(c)}^{i}\left[\left(\prod_{v'\in\mathcal{N}(c)/v}\operatorname{sgn}\left|x_{e_{v',c}^{H}}^{Hd_1}\right|\right)\cdot\max\left(0,\right.\right.$$
$$\left.\left.\alpha_{dc(c)}^{i}\cdot\min_{v'\in\mathcal{N}(c)/v}\left|x_{e_{v',c}}^{Hd_1}\right|+\beta_{dc(c)}^{i}\right)\right]. \quad (30)$$

It is noteworthy that if we consider highly popular check-regular LDPC codes[3], only variable-dimensional scaling is needed, and we can use the same quantization for both variable nodes and check nodes within an iteration. So, it is not necessary to assign learnable parameters in this layer. Thus, the computational complexity can be reduced by cutting down the number of trainable variables. Since the layer applies only minimization and symbolization operations, the messages outgoing from this layer have the same value as the incoming messages. We can omit the quantization in this layer, and the outputs are:

$$x_{e_{v,c}}^{Hd_2} = \prod_{v'\in\mathcal{N}(c)/v}\operatorname{sgn}\left|x_{e_{v',c}}^{Hd_1}\right|\cdot\min_{v'\in\mathcal{N}(c)/v}\left|x_{e_{v',c}}^{Hd_1}\right|. \quad (31)$$

The output layer represents the final hard decision step in the BP decoder and outputs the final marginalization. This layer has N neurons and the same connection to the previous layer as the second layer. It also receives irrelevant information from the concatenated NOLD decoder. The outputs of this layer are defined as:

$$x_v^{Ot} = l_v^0 + \sum_{c\in\mathcal{M}(v)} x_{e_{v,c}}^{Hd_{2i}}. \quad (32)$$

The outputs are N-length messages associated with the NOLD decoding outputs at iteration $i+1$. Before calculating the loss function, the outputs of the final layer should be preprocessed by an activation function $\theta(\cdot)$ in (26):

$$o_v = \theta\left(x_v^{Ot}\right). \quad (33)$$

The network renews the parameters by minimizing the gap between the final outputs and target codewords. When the neural optimization stops, we obtain the optimized decoding parameters for the ith iteration. For the next iteration, and $(i+1)$-iteration NOLD decoder is fed by the newly optimized parameters and outputs the training database for the initialized NOLD network. Then the network optimizes the parameters for the $(i+1)$th iteration. The neural optimization for the parameters carries on until the maximum number of iteration is reached.

E. Discussions and Complexity Analysis

In the proposed NOLD algorithm, both the scaling and quantization parameters contribute to the LDPC decoding performance, but in very different ways. The scaling factors can narrow the MS messages to approximate the SP messages, which is beneficial for the decoding accuracy. At the same time, the low-resolution quantization reduces the memory cost and hardware complexity. To identify the contribution of scaling factors under low-resolution, we proposed two strategies for optimizing the NOLD decoder's decoding parameters.

1) Joint Optimization. During the training stage, with such a strategy, both the scaling factors and the quantization parameters are jointly assigned to and optimized by the proposed network.

2) Single Optimization. We only optimize the quantization parameters by the proposed network off-line, thus yielding low computational complexity. The scaling factors are set to $\alpha = 1.0$ and $\beta = 0.0$. See Section V.D for more explanations.

For the long-length codewords, a large number of parameters in the standard DNN lead to high computational complexity and strongly restrict the usability in budget-constrained devices even though for a four-layer shallow network. However, the proposed method introduced a quantization method that reduces memory usage or complexity during the learning phase while suffering from a minimal drop-in decoding performance compared to floating baselines. Besides, efficient implementations can benefit from the check-to-variable layers in which simple arithmetical multiplications follow complex hyperbolic tangent operations. Furthermore, The sharing weight for the nodes with the same degree reduces the number of network parameters.

It is noted that when comparing to the classical MS decoder with belief messages, the proposed method employed the quantized messages, which should be optimized by a neural network. The proposed method achieves some performance gain at the expense of an increased training budget. Let $O(N_{LDPC})$ be the number of floating-point operations (FLOPs) by the classical LDPC decoder. As shown in Fig. 2, the proposed method has an additional training complexity from a four-layer neural network and some numeric operations (the minimization shown in Eq. (31)). Here, we borrow the analytic tools from [40] to give detailed complexity analysis (indicated by FLOPs). For one Multilayer Perceptron based network layer, we can compute the FLOPs [40] as

$$FLOPs = O\left(\left(2D_{in}-1\right)D_{out}\right), \quad (34)$$

where D_{in} and D_{out} are the input and output dimensionality, respectively. In out case, with the analysis shown in Section IV-D, we have $FLOPs = O\left(\left(2E-1\right)\left(3E+D\right)\right)$, where E and N denote the numbers of edges and nodes, respectively. Overall, for the training stage, with K iterations, the proposed method will enjoy $FLOPs = O\left(K\left(\left(2E-1\right)\left(3E+D\right)\right)+E\right)$ over the well-established LDPC decoder (i.e., MS). However, for the Online inference, we only need to send the trained parameters (scaling factors and the quantization parameters) to the decoder, which indicates that the proposed method can have a similar computational complexity with the classical decoder.

V. Simulations

In all experiments, the simulations are implemented in the TensorFlow platform. The experimental LDPC codes are $(1998, 1512)$ quasi-cyclic LDPC code and $(155, 62)$ Tanner code taken from [41] and [42]. The two LDPC codes are labeled as I [$(1998, 1512)$ quasi-cyclic LDPC code] and II [$(155, 62)$ Tanner code] for future reference.

[3]The check-regular LDPC codes are highly popular since they are capacity-achieving in a much stronger sense than other LDPC codes and have a more straightforward implementation in hardware.

Before training the network, the training data is first generated under AWGN. In the following experiments, we use all-zero codewords. We pay particular attention to the range of SNR that corresponds to the BER $= 10^{-4} \sim 10^{-5}$ and its proximity since it's the critical SNR range in LDPC decoder implementations. The training data is generated under multiple SNRs in the critical SNR range(9 SNRs for a 2dB-SNR range). For each SNR, we generate 4×10^8 bits of LDPC codes as training data. Another 2×10^7 bits of LDPC codes are generated as the validation data to achieve 100 codeword errors at the BER level of $10^{-4} \sim 10^{-5}$.

The proposed network is set to two-bit resolution (i.e, $b = 2$). The initial value of the quantization parameter(i.e. quantization step Δ) is set to $\frac{2}{3}$ and the initial quantization level set is set to $\{-1, -1/3, 1/3, 1\}$. As for the scaling factors, the initial values of the normalizing factor and offset factor are respectively set to $\alpha = 1.0$ and $\beta = 0.0$.

We use the stochastic gradient descent during the network training to train the neural network with a training batch size of 1000. We use the Adam optimization method to search for the optimal decoding parameters. The loss will be checked every 100 epochs over the validation set, and the training process will reach an early stop if the loss does not drop for some time(800 epochs in our experiment).

Several state-of-art algorithms, such as the numerical method [11], the adaptive method [28] and the DE method [16] are included for decoding performance comparison. In detail, the adaptive method adds an adaptive module to select correct decoding parameters for the next iteration after each decoding iteration. The DE method computes the decoding error probability and searches for the decoding parameters that minimize the error probability. The MS decoder with infinite-resolution data converters serves as the benchmark for all compared methods. The maximum number of decoding iterations is fixed to 20. The decoding performance is measured by the coded bit error rate (BER). The above settings are the same throughout all experiments unless otherwise specified.

A. The Decoding Performance Over Regular Codes And Irregular Codes

We first compare the 2-bit decoding performance of the proposed algorithm and other state-of-art algorithms under AWGN noise. For the numerical method in [11], the parameters are optimized in a specific SNR range for all SNR values and utilize the optimized parameters of the first iteration for all subsequent iterations. The adaptive method in [28] adopts the three sets of parameters and chooses one set of parameters from these three sets according to an adaptive rule for different iterations and SNRs. The DE decoder in [16] is also included for comparison. We present the results of all employed methods for three BER regions (labeled as "high BER region", "medium BER region", "low BER region"). Three BER regions represent a certain BER range that is respectively corresponding to: $10^{-2} \sim 10^{-3}$, $10^{-3} \sim 10^{-4}$, and $10^{-4} \sim 10^{-5}$. The BER performance of all the mentioned methods for the irregular code I and the regular code II is shown in Fig.3 and Fig.4, respectively.

In Fig.3, the performance of the numerical method with three BER regions has a similar decoding performance to the adaptive method in their respective SNR range but can not achieve ideal performance out of that SNR region. This can be explained by the fact that the numerical method and the adaptive method use fixed parameters for the irregular LDPC codes. In fact, no matter what parameters the decoder chooses, the fixed parameters can not efficiently improve the decoding performance over a wide range of SNRs. On the other hand, as shown in Fig.3, the adaptive decoder in [28] can support a wide range of SNRs. The DE method that uses different parameters for different iterations and SNRs can also get a good performance close to the adaptive method. The significant performance gains of both the adaptive method and the DE method suggest that we should apply dynamic decoding parameters for irregular LDPC codes. Our proposed NOLD decoder dynamically updates the parameters updating strategy with a neural network optimization method and achieves better performance than all competing methods, especially when BER is between 10^{-3} and 10^{-5}.

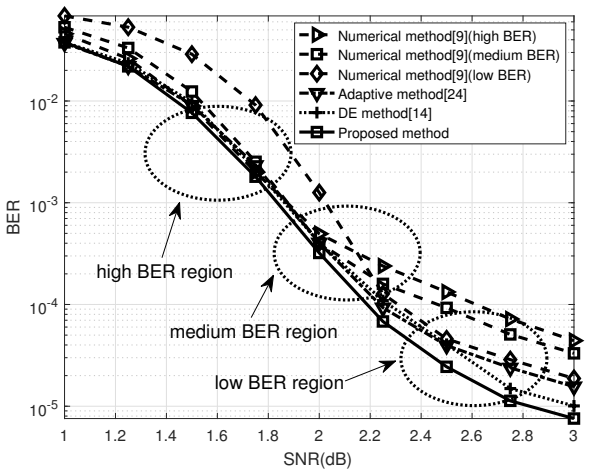


Fig. 3. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under two-bit quantization over the AWGN channel.

For the regular code II, we present the results of the numerical method for the SNR range from 0.5dB to 1.0dB that is corresponding to BER from 10^{-4} to 10^{-5} in Fig.4. For the regular LDPC codes, all decoders' performance improves linearly with the increased SNR, demonstrating that the decoding performance is less sensitive to the decoding parameters compared with the irregular LDPC codes. Fig.4 also illustrates that the adaptive method and the DE method have similar decoding performance and achieve a slight performance gain compared with the numerical method. The NOLD algorithm slightly outperforms the DE method and obtains comparable performance with the floating MS algorithm.

Overall, the proposed method based on deep learning optimization strategy can get remarkable performance improvement over other compared methods for irregular LDPC codes and attain comparable performance with the floating MS algorithm in the regular LDPC codes with a wide range of SNRs. The results shown in Fig.3 and Fig.4 demonstrate the feasibility of low-resolution decoders and indicate the superiority of the proposed method.

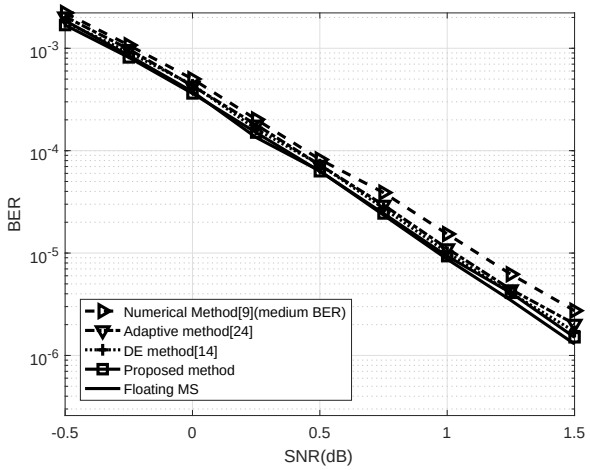


Fig. 4. The BER performance of the $(155, 62)$ Tanner code under two-bit quantization over the AWGN channel.

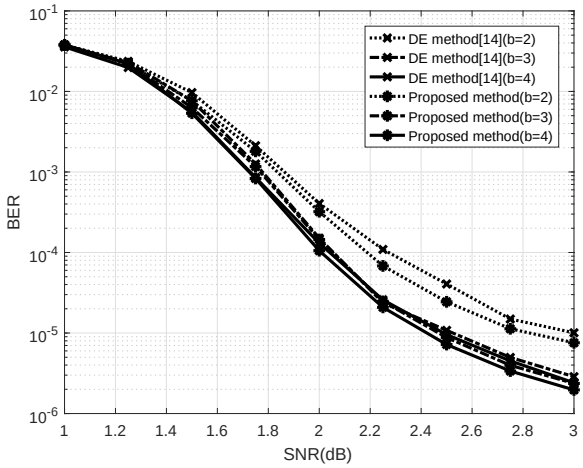


Fig. 5. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under different low-resolution quantization over the AWGN channel.

B. Effect of Quantization Level

The quantization resolution is an important parameter that balances the trade-off between decoding accuracy and hardware cost. In hardware realization, the common quantization resolution values vary from 4 bits to 7 bits [30]. We will focus on the low-resolution case. In Fig.3 and Fig.4, we have presented the results of decoding performance under 2-bit resolution ($b = 2$), which show the significant performance gain of the proposed NOLD algorithm over all competing decoders. To illustrate the effect of the low-resolution quantization, we further report the simulation results under 3-bit ($b = 3$) and 4-bit ($b = 4$) resolutions in Fig.5.

It is shown in Fig.5 that the NOLD algorithm can achieve a noticeable performance gain compared with the adaptive method and the DE method at both 3-bit and 4-bit resolutions. At the 3-bit resolution, the NOLD method can achieve about 0.2-dB performance gain over the adaptive method and about 0.1-dB performance gain over the DE method when BER is 10^{-5}. At the 4-bit resolution, the proposed algorithm can also improve the decoding performance to the same extent as the 3-bit resolution compared with the adaptive method and the DE method. The performance gap between the 4-bit resolution and the floating one (full-resolution) is almost negligible, indicating the feasibility of the low-resolution decoders.

C. Effect of Different Channel Noises

So far above simulation results are obtained under AWGN. The proposed algorithm will not be limited to a specific noise model. This should be rational since the neural network can learn to deal with different kinds of noise. The proposed algorithm remains effective under other noises by re-training the network. To verify this, we consider another two noise models as depicted in Sections II.B.

C.1 Rayleigh Fading Noise

In Fig.6, we show the 2-bit decoding performance of all the employed methods over the Rayleigh fading noise channel. The numerical method that chooses the best parameters in the SNR range corresponding to BER from 10^{-4} and 10^{-5} (labeled as "medium SNR") can not successfully deal with the Rayleigh fading noise and suffers a 0.2-dB performance loss compared with the adaptive method. A performance gain of 0.2 dB compared to the adaptive approach is observed over the DE method. We can also see that the NOLD algorithm and the DE method have similar performance for SNR lower than 9dB. When SNR is higher than 9 dB, the NOLD algorithm can achieve about 0.25 dB performance gain over the DE method.

We execute the 3-bit and 4-bit decoding for the employed methods and compare their decoding performance under the same SNR range as the 2-bit resolution. The low-resolution decoding performance under Rayleigh fading noise is shown in Fig.7. At the 3-bit resolution, the DE method has comparable performance with the adaptive method. The NOLD algorithm can obtain about 0.2 dB performance at BER $= 10^{-5} \sim 10^{-6}$ compared with the two methods. At the 4-bit resolution, the DE method achieves about 0.2 dB performance gain over the adaptive method. Our proposed algorithm significantly outperforms the compared methods and achieves comparable performance with the floating MS decoder when SNR is $8 \sim 9$dB.

C.2 Additive Correlated Gaussian Noise

In Fig.8, we report the simulation results under ACGN with a strong correlation ($\varphi = 0.8$) at the 2-bit resolution for all considered methods. It is shown in Fig.8 that the adaptive method can achieve about 0.1 dB performance gain over the numerical method for the SNR range that corresponds to BER from 10^{-4} to 10^{-5} (labeled as "medium SNR"). The DE method outperforms the adaptive method with about 0.25 dB gain at the same SNR range and can achieve comparable performance with the proposed NOLD algorithm.

We also present the results of the 3-bit and 4-bit resolution under strong correlation ($\varphi = 0.8$) in Fig.9. It should be noted that the tested SNRs are from 2.5dB to 4.5dB which are the same as the 2-bit resolution. Under the tested SNRs, the DE method performs better than the adaptive method but is worse than the proposed NOLD algorithm. The proposed method can reach a

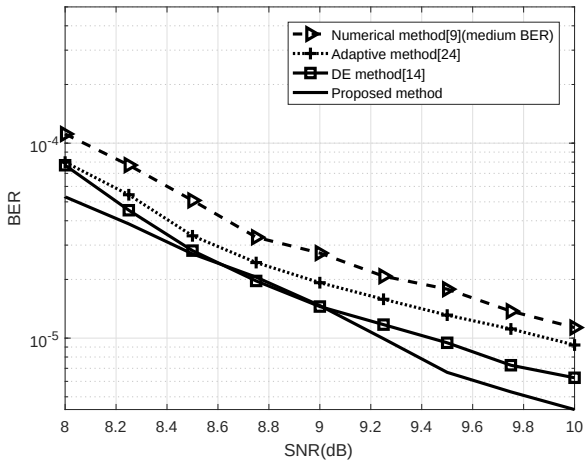


Fig. 6. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under two-bit quantization over the Rayleigh fading channel.

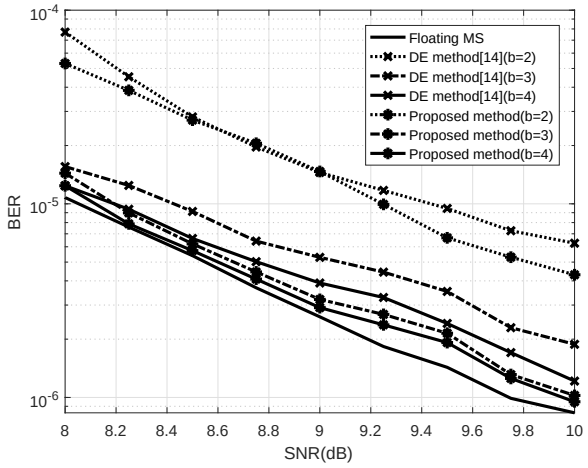


Fig. 7. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under low-resolution quantization over the Rayleigh fading channel.

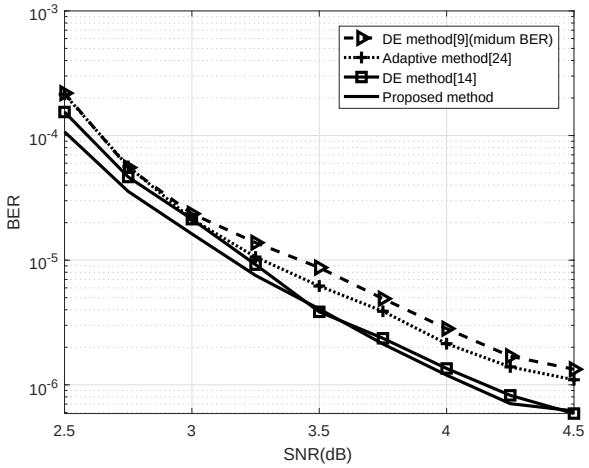


Fig. 8. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under two-bit quantization over the ACGN channel.

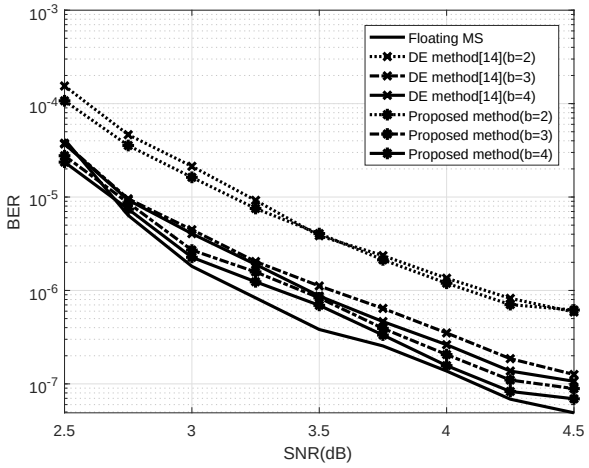


Fig. 9. The BER performance of $(1998, 1512)$ quasi-cyclic LDPC code under low-resolution quantization over the ACGN channel.

performance near the floating MS decoder, suggesting that the 4-bit resolution is favored for implementation.

Overall, the proposed decoder has robust performance in all considered cases, confirming that the proposed algorithm can support different noises.

D. The Decoding Performance Under Joint Optimization And Single Optimization

In this subsection, we report the simulation results in Fig.10, using different optimization strategies for the NOLD algorithm and the DE method. It is shown in Fig.10 that the NOLD algorithm can outperform the DE method for both joint optimization and single optimization. The joint optimization delivers better performance than the single optimization at the cost of higher computational complexity. This performance gain is about 0.1dB for the NOLD algorithm and 0.05dB for the DE method when BER is $10^{-4} \sim 10^{-5}$dB. It is suggested that the proposed method with single optimization can be adopted for practical implementation since the quantization parameters can be optimized off-line.

E. Effect of Decoding Iteration

We have only presented the results of the proposed NOLD algorithm with 20 iterations. However, the iteration number is designed to be as low as possible in a low-latency scenario. We first set the iteration number to $I = 5$ and then compare the BER performance of all employed methods. We also plot the performance of 15 iterations and 20 iterations for comparison. As shown in Fig.11 increasing the decoding iterations can enhance the performance gain, and 20 iterations can improve the decoding performance by 0.6 dB at BER $= 10^{-4}$ compared with five iterations. Besides, we also notice that after 15 decoding iterations, the performance improvement becomes smaller. The reason is that the decoder has almost reached its maximum capacity and cannot further correct the error codewords.

At the low iteration number ($I = 5$), the adaptive method can get similar performance to the DE method in SNR range from 1dB to 3dB. And the NOLD algorithm has a significant per-

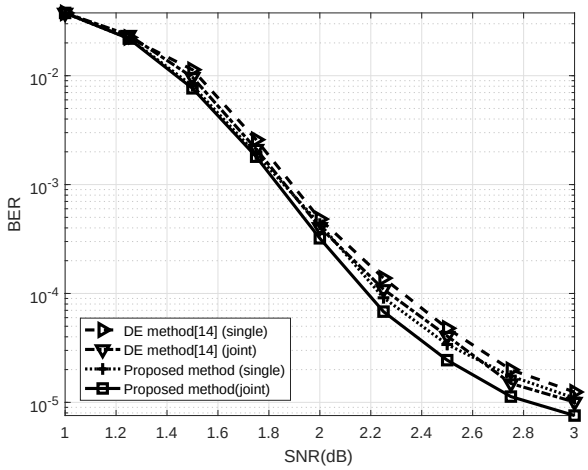


Fig. 10. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under joint optimization and single optimization over the AWGN channel.

formance gain over these decoding methods. The same results can be observed for $I = 15$ and $I = 20$. Although it is unknown whether such an optimal local solution of each iteration is guaranteed to be globally optimal, our proposed method can accelerate the decoder's convergence speed and get remarkable performance regardless of the iteration number. Thus we conclude that it is efficient to replace the end-to-end optimization algorithm with our proposed NOLD algorithm.

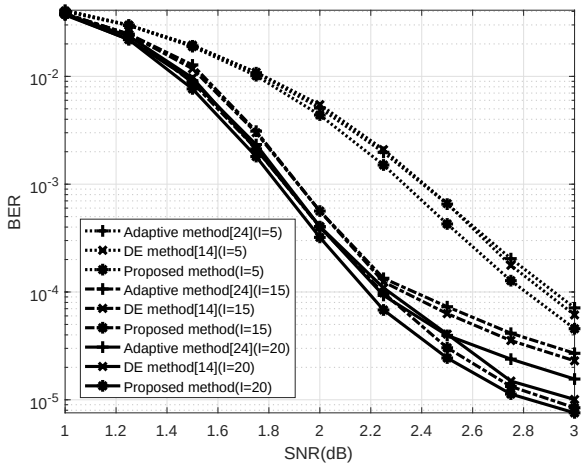


Fig. 11. The BER performance of the $(1998, 1512)$ quasi-cyclic LDPC code under 5, 15 and 25 iterations over the AWGN channel.

VI. Conclusions

This paper has introduced the modified MS decoding scheme, which operates scaling and quantization on different variable-node and check-node messages. The NOLD algorithm based on a deep learning optimization strategy has been proposed to optimize decoding parameters. The introduced neural quantizer can handle the discrete variables optimization problems incurred by the low-resolution quantization. Various experiments have been conducted to verify the performance of the proposed method. For both regular and irregular LDPC codes, the experimental results demonstrate that the proposed NOLD algorithm outperforms all competing techniques with various conditions, i.e., quantization levels, noises, and iterations. For our future work, we will investigate the proposed decoder over the intelligent reflecting surface assisted multiple-input multiple-output (MIMO) systems, which is foreseen to be the critical enabler of the future communication systems [43]–[45].

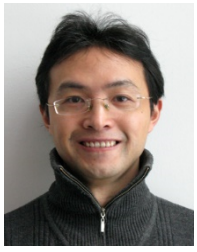
REFERENCES

[1] R. Gallager, "Low-density parity-check codes," IRE Trans. Inf. Theory, vol. IT-8, pp. 21-28, 1962.
[2] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," IEEE Trans. Inform. Theory, vol. 47, no. 2, pp. 599-618, 2001.
[3] R. Maunder, "The 5G channel code contenders," White Paper, Accelercomm Ltd., 2016.
[4] R. Qiu and M. Wicks, Cognitive Networked Sensing and Big Data. New York, NY, USA: Springer, 2013.
[5] H., Chen, et al. , "Iterative reliability-based modified majority-logic decoding for structured binary LDPC codes.", J. Commun. Netw., vol. 17, no. 4, pp. 339-345, 2015.
[6] Ankan, Erdal, et al. , "Challenges and some new directions in channel coding.", J. Commun. Netw., vol. 17, no. 4, pp. 328-338, 2015.
[7] M. Fossorier, M. Mihaljevic, and H. Imai, "Reduced complexity iterative decoding of low-density parity check codes based on belief propagation," IEEE Trans. Commun., vol. 47, pp. 673-680, 1999.
[8] J. Chen, A. Dholakia, E. Eleftheriou, M. Fossorier, and X. Hu, "Reduced-complexity decoding of LDPC codes," IEEE Trans. Commun., vol. 53, no. 8, pp. 1288-1299, 2005.
[9] J. Chen and M. Fossorier, "Decoding low-density parity-check codes with normalized APP-based algorithm," in Proc. IEEE GLOBECOM, pp. 1026-1030, 2001.
[10] J. Chen and M. Fossorier, "Density evolution for two improved BP Based decoding algorithms of LDPC codes," IEEE Commun. Lett., vol. 6, no. 5, pp. 208-210, 2002.
[11] J. Chen and M. Fossorier, "Near optimum universal belief propagation based decoding of low-density parity check codes," IEEE Trans. Commun., Vol. 50, No. 3, pp. 406-414, 2002.
[12] G. Lechner and J. Sayir, "Improved sum-min decoding for irregualr LDPC codes," in Proc. IEEE Intern. Symp. on Turbo Codes and related topics, 2006.
[13] L. Fan, C. Pan, K. Peng, and J. Huang, "Adaptive normalized min-sum algorithm for LDPC decoding," in Proc. 9th Int. Wireless Commun. Mobile Comput. Conf., pp. 1081-1084, 2013.
[14] A. Emran and M. Elsabrouty, "Simplified variable-scaled min sum LDPC decoder for irregular LDPC codes," in Proc. CCNC, pp. 518-523, 2014.
[15] H. Zhao, K. Zhang and M. Shi, "Adaptive reliability-based iteration min-sum decoding algorithm for majority-logic decodable LDPC codes," in Proc. ICCCNT, pp. 1-4, 2015.
[16] J. Heo and K. Chugg, "Optimization of scaling soft information in iterative decoding via density evolution methods," in IEEE Trans. Commun., vol. 53, no. 6, pp. 957-961, 2005.
[17] J. Zhang, M. Fossorier, and D. Gu, "Two-dimensional correction for min-sum decoding of irregular LDPC codes," in IEEE Commun. Lett., vol. 10, no. 3, pp. 180-182, 2006.
[18] B. S. Tan, K. H. Li and K. C. Teh, "BER analysis of LDPC codes with generalized selection combining over a Rayleigh-fading channel using Gaussian approximation", IET Commun., vol. 6, no. 1, pp. 90-96, 2012.
[19] Z. Mei, K. Cai, and G. Song, "Performance analysis of finite-length LDPC codes over asymmetric memoryless channels, " in IEEE Trans. Veh. Technol., vol. 68, no. 11, pp. 11338-11342, 2019.
[20] L. Chu, F. Wen and R. C. Qiu, "Eigen-Inference Precoding for Coarsely Quantized Massive MU-MIMO System With Imperfect CSI," in IEEE Trans. on Veh. Technol., vol. 68, no. 9, pp. 8729-8743, Sept. 2019.
[21] C. Studer, N. Preyss, C. Roth, and A. Burg, "Configurable highthroughput decoder architecture for quasi-cyclic LDPC codes," in Proc. 42nd Asilomar Conf. Signals, Syst., pp. 1137-1142 Comput., 2008.
[22] L. Chu, F. Wen, F. Li, and R. C. Qiu "Efficient nonlinear precoding for massive MIMO downlink systems with 1-bit DACs," IEEE Trans. on Wireless Commun., vol.18, pp. 4213-4224, 2019.
[23] S. Sharma, S. Chatzinotas, and B. Ottersten, "SNR estimation for multi-dimensional cognitive receiver under correlated channel/noise," IEEE Trans. Wireless Commun., vol. 12, no. 12, pp. 6392-6405, 2013.
[24] D. Oh and K. Parhi, "Performance of quantized min-sum decoding algo-

rithms for irregular LDPC codes," in Proc. IEEE ISCAS 2007, New Orleans, LA, pp. 2758-2761, 2007.
[25] E., Kavvousanos, "Deep-learning-based forward-error-correction decoding techniques and optimizations for hardware implementation," Master Thesis, University of Patras, 2020.
[26] X. Zhang and P. Siegel, "Quantized iterative message passing decoders with low error floor for LDPC codes," IEEE Trans. Commun., no. 1, pp. 1-14, 2014.
[27] V., Bane, and X. Xiao, and S. Lin, "Learning to decode LDPC codes with finite-alphabet message passing," Inform. Theo. and App. Workshop (ITA), IEEE, 2018.
[28] H. He, L.Chu, and R. Qiu, "A New Low-Resolution Min-Sum Decoder Based on Dynamic Clipping for LDPC Codes," in Proc. IEEE Int. Conf. Commun. China., 2019.
[29] J. Chen and P. Fossorier, "Density evolution for BP-based decoding algorithms of LDPC codes and their quantized versions," in Proc. IEEE GLOBECOM, vol. 2., pp. 1378-1382, 2002.
[30] A. Balatsoukas-Stimming and A. Burg, "Density Evolution for Min-Sum Decoding of LDPC Codes Under Unreliable Message Storage," IEEE Commun. Lett., vol. 18, no. 5, pp. 849-852, 2014.
[31] Z. Mheich, T. Nguyen-Ly, V. Savin, and D. Declercq, "Code-aware quantizer design for finite-precision min-sum decoders," in IEEE BlackSeaCom, Varna, Bulgaria, 2016.
[32] A. Zappone, M. Renzo, and M. Debbah, "Wireless Networks Design in the Era of Deep Learning: Model-Based, AI-Based, or Both?," arXiv:1902.02647.
[33] S., Salih, et al. , "Intelligent network data analytics function in 5G cellular networks using machine learning.", J. Commun. Netw., vol. 22, no. 3, pp. 269-280, 2020.
[34] X. Wu, M. Jiang, and C. Zhao, "Decoding Optimization for 5G LDPC Codes by Machine Learning," IEEE Access, vol. 6, pp. 50179-50186, 2018.
[35] E. Nachmani, E. Marciano, L. Lugosch, W. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," IEEE J. Sel. Topics Signal Process., vol. 12, no. 1, pp. 119-131, 2018.
[36] Aurich R., Steiner F., "Staircase functions, spectral rigidity, and a rule for quantizing chaos," Phys. Rev. A, vol. 45, no. 2, pp.583-592, 1992.
[37] G . Bachmann, "Fourier and wavelet analysis", Sprihger-Verlag, 2000.
[38] T. Wadayama and S. Takabe, "Quantizer Optimization Based on Neural Quantizer for Sum-Product Decoder", IEEE GLOBECOM, pp. 1-6, 2018.
[39] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," IEEE J. Sel. Topics Signal Process., vol. 12, no. 1, pp. 144-159, 2018.
[40] A. Canziani, A. Paszke, E. Culurciello, "An analysis of deep neural network models for practical applications," arXiv preprint arXiv:1605.07678, 2016.
[41] UCLA CSL Published Codes and Design Tools, online: http://www.seas.ucla.edu/csl/#/publications/published-codes-and-design-tools
[42] M. Helmling and S. Scholl, "Database of channel codes and ML simulation results," Wimax 3/4A, University of Kaiserslautern, Kaiserslautern, Germany, 2016. [Online]. Available: www.uni-kl.de/channel-codes.
[43] Di Renzo M, Debbah M, Phan-Huy D T, et al. "Smart radio environments empowered by reconfigurable AI meta-surfaces: an idea whose time has come," EURASIP J. Wireless Commun. Netw., 2019.
[44] L. Chu, H. He, L. Pei, et al. "Label-free Optimization for Passive Beamforming in IRS-assisted MISO Systems" IEEE 20th International Conf. on Comm. Tech., pp. 157-162, 2020.
[45] Q. Wu, R. Zhang. "Towards smart and reconfigurable environment: Intelligent reflecting surface aided wireless network[J]. IEEE Comm. Mag., 2019, vol. 58, no. 1, pp. 106-112.

Lei Chu (Member, IEEE) is a senior research fellow at the School of Electronics, Information and Electrical Engineering, Shanghai Jiao Tong University (SJTU), where he defended his Ph.D. degree in Dec. 2019. He was a visiting scholar at the University of Tennessee, Knoxville, in 2019. He has authored three book chapters and over twenty papers in many refereed journals, such as IEEE T-WC, T-VT, T-PS, and T-BD. His current research interests are broadly in high-dimensional statistics, numerical optimization, and neural network optimization, as well as their applications in wireless communications and smart healthcare. He serves as a regular technical reviewer for many journals, such as IEEE Transactions on Signal Processing, IEEE Transactions on Big Data, IEEE Transactions on Multimedia, IEEE Transactions on Communications, IEEE Transactions on Vehicular Technology, and IEEE Wireless Communications Magazine. He received the Excellent Research Award for SJTU overseas study in Jan. 2019 and the outstanding Ph.D. Graduate Award of SJTU in Mar. 2020, and International Postdoctoral Exchange Fellowship in Jun. 2020. He is dedicated to reproducible research and has made a great number of codes publicly available.

Huanyu He received the B.E. and M.E. degrees of Communication Engineering from Nanjing University in 2016 and Shanghai Jiao Tong University in 2020. He is currently pursuing a Ph.D. in the Department of Electronic Engineering at Shanghai Jiao Tong University. His current research interests include wireless communication and deep learning.



Ling Pei (Senior Member, IEEE) received the Ph.D. degree from Southeast University, Nanjing, China, in 2007. From 2007 to 2013, he was a Specialist Research Scientist with the Finnish Geospatial Research Institute. He is currently a Professor with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University. He has authored or co-authored over 90 scientific papers. He is also an inventor of 24 patents and pending patents. His main research is in the areas of indoor/outdoor seamless positioning, ubiquitous computing, wireless positioning, Bio-inspired navigation, context-aware applications, location-based services, and navigation of unmanned systems. Dr. Pei was a recipient of the Shanghai Pujiang Talent in 2014.

Robert C. Qiu (IEEE S'93-M'96-SM'01-F'14) received the Ph.D. degree in electrical engineering from New York University. He was a Professor in the Department of Electrical and Computer Engineering, Center for Manufacturing Research, Tennessee Technological University, Cookeville, Tennessee, where he started as an Associate Professor in 2003 before becoming a Professor in 2008. From 2015, he joined Shanghai Jiao Tong University as a Zhiyuan Endowed Professor. He was Founder-CEO and President of Wiscom Technologies, Inc., manufacturing and marketing WCDMA chipsets. He holds over 20 patents and authored over 200 technical papers. He has 15 contributions to 3GPP and IEEE standards bodies. In 1998, he developed the first three courses on 3G for Bell Labs researchers. He served as an adjunct professor at Polytechnic University, Brooklyn, New York. He served as editor for many peer-reviewed journals. His current interests are random matrix theory-based theoretical analysis for deep learning and wireless communication applications.