IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

SPECIAL SECTION ON ARTIFICIAL INTELLIGENCE FOR PHYSICAL-LAYER WIRELESS COMMUNICATIONS

Received January 13, 2020, accepted January 27, 2020, date of publication February 4, 2020, date of current version February 12, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2971526*

# Learning to Decode Polar Codes With One-Bit Quantizer

**JIAN GAO, (Student Member, IEEE), KAI NIU, (Member, IEEE), AND CHAO DONG, (Member, IEEE)**

Key Laboratory of Universal Wireless Communications, Ministry of Education, Beijing University of Posts and Telecommunications, Beijing 100876, China

Corresponding author: Kai Niu (niukai@bupt.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFE0205501, and in part by the National Natural Science Foundation of China under Grant 61671080.

**ABSTRACT** A deep learning method for improving the performance of polar belief propagation (BP) decoder equipped with a one-bit quantizer is proposed. The method generalizes the standard polar BP algorithm by assigning weights to the layers of the unfolded factor graph. These weights can be learned autonomously using deep learning techniques. We prove that the improved polar BP decoder has a symmetric structure, so that the weights can be trained by an all-zero codeword rather than an exponential number of codewords. In order to accelerate the training convergence, a layer-based weight assignment scheme is designed, which decreases the amount of trainable weights. Simulation results show that the improved polar BP decoder with a one-bit quantizer outperforms the standard polar BP decoder with a 2-bit quantizer and achieves faster convergence.

**INDEX TERMS** Polar codes, one-bit decoder, deep learning, BP algorithm.

## I. INTRODUCTION

Polar codes are a novel channel coding technique proposed by Arıkan [1] and have been proven to achieve the capacity of binary-input discrete memoryless channels (B-DMCs) under the successive cancellation (SC) decoding. In addition, polar codes are also of interest because of their low encoding and decoding complexity. Under the CRC-aided SC list (CA-SCL) decoding [2]–[4], polar codes with finite code length achieve competitive performance to turbo codes or low-density parity-check (LDPC) codes. Therefore, polar codes have been selected as the coding scheme for the control channel of enhanced mobile broadband (eMBB) scene in the 5G communication system [5].

In the implementation of modern receivers that require high data rates, the use of one-bit ADCs can reduce the power consumption [6]. In light of this, one-bit quantized massive multiple-input multiple-output (MIMO) and orthogonal frequency division multiplexing (OFDM) systems are studied [7]–[9]. However, in applications such as the Internet of Things, cyber-physical systems or wireless sensor networks, low-delay transfer of analog measurements is a more relevant task for high-rate communication [10], [11]. Therefore, we focus on the zero-delay transmission model on the additive white Gaussian noise (AWGN) channel in presence of a one-bit quantizer (Fig. 3), and the polar decoder is optimized in this paper.

One-bit quantization introduces strong nonlinearities and other intractable features that render conventional polar decoder far from optimal. Motivated by the success of deep learning in many different challenging communication tasks [12]–[15], this technique is considered to compensate for the loss of polar decoding performance caused by the one-bit quantizer. In the existing neural decoding research, Gruber et al. proposed to use a deep feedforward neural network decoding random codes and polar codes in [16]. However, the exponential training complexity makes it only suitable for short code-length. A series of model-driven neural network decoders are proposed in [17]–[23], which can be trained with an all-zero codeword. They are built on the standard BP algorithm and represented as BP-NNs. Unfortunately, BP-NNs lack a theoretical basis and cannot maintain fast training convergence in the zero-delay model equipped with a one-bit quantizer.

In this paper, the weighting scheme of the weighted polar BP algorithm is optimized to accelerate training convergence. In addition, we investigate the symmetric structure of the BP-NNs, which provide an interpretive theory for the

The associate editor coordinating the review of this manuscript and approving it for publication was Guan Gui.

property of BP-NNs that can be trained by an all-zero codeword. The main contributions of this paper are summarized as follows:

1) Our decoder is constructed by assigning weights to the edges of the unfolded polar BP factor graph and uses the weights to adjust the results of message updates. The introduced weights provide additional optimizable parameters, and the standard polar BP algorithm is a special form in which all of the weights are one. A layer-based weighting scheme is proposed to guide the learning process of the weights, and the small number of trainable weights leads to faster learning than conventional BP-NNs.
2) We investigate the symmetrical structure of the BP-NNs. Under the condition of symmetry, the whole behavior of the BP-NNs can be predicted from the behavior of assuming that an all-zero codeword is transmitted. Therefore, it is sufficient to train the BP-NNs with an all-zero codeword. This paper provides an interpretive theory for the property of BP-NNs that can be trained by an all-zero codeword.

The remainder of the paper is organized as follows. In Section II, basic concepts of polar codes and the polar BP algorithm are reviewed. Section III presents the system model with the one-bit quantizer. In section IV, the architecture and weighting scheme of the layer-based weighting BP-NN (LBP-NN) are introduced. We prove the symmetrical structure of the LBP-NN in section V. Section VI gives the simulation results. Finally, Section VII concludes the works.

Throughout the paper, calligraphic characters are used to denote sets, such as $\mathcal{X}$. The $x$, $\mathbf{x}$, and $\mathbf{X}$ denote scalar, vector, and matrix, respectively.

## II. PRELIMINARIES

In this section, we review the basic concepts of polar codes and introduce the standard polar BP algorithm.

### A. POLAR CODING

Polar codes proposed in [1] are constructed based on the kernel matrix $\mathbf{F}_2 = \left[\begin{smallmatrix}1 & 0\\ 1 & 1\end{smallmatrix}\right]$. For an $(N, K)$ polar code with length $N = 2^n$, the most reliable $K$ polarized subchannels are selected to transmit the information bits, and the set $\mathcal{A}$ is the collection of information bit indices. The remaining $N-K$ subchannels are assigned to carry fixed values which are usually set to zeros. The set $\mathcal{A}^c$ corresponds to the collection of frozen bit indices. The generator matrix of the $N$-length polar code is defined as $\mathbf{F}_N = \mathbf{F}_2^{\otimes n}$, where $\mathbf{F}_2^{\otimes n}$ represents the $n$-th Kronecker power of $\mathbf{F}_2$. An $N$-length codeword $\mathbf{x}$ is calculated by

$$\mathbf{x} = \mathbf{u}\mathbf{F}_N, \tag{1}$$

where the source vector $\mathbf{u} = \{u_0, u_1, \ldots, u_N\}$ is generated by assigning $u_i = 0$ for $i \in \mathcal{A}^c$, and the remaining elements are assigned as information bits.

Consider an AWGN channel and binary phase shift keying (BPSK) modulation, the channel output vector $\mathbf{y}$ corresponding to the transmitted codeword $\mathbf{x}$ is

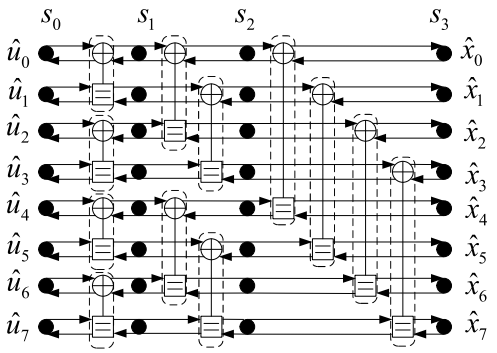



**FIGURE 1.** The polar BP factor graph of (8, 4) polar codes with $\mathcal{A}^c = \{0, 1, 2, 4\}$.

$$\mathbf{y} = (\mathbf{1}^N - 2\mathbf{x}) + \mathbf{z}, \tag{2}$$

where $\mathbf{1}^N$ is a all-one vector with size $N$ and $\mathbf{z} = \{z_0, z_1, \ldots, z_{N-1}\}$ is the AWGN vector with $z_i \sim \mathcal{N}(0, \sigma^2)$. The log likelihood ratio (LLR) vector that computed from the channel output vector $\mathbf{y}$ is

$$L_i = \log \frac{\Pr(x_i = 0|y)}{\Pr(x_i = 1|y)} = \frac{2y_i}{\sigma^2}, \quad i = 0, 1, \ldots, N-1. \tag{3}$$

### B. BP DECODING OF POLAR CODES

The polar BP algorithm can be performed over a factor graph, as illustrated in Fig. 1, where the message nodes are represented by black circles, and the processing elements (PEs) are represented in dotted rectangles. The messages are updated through the PEs and iteratively propagated. A complete iteration begins with a right-to-left message transmission that propagates the LLR from the channel side (rightmost) to the source side (leftmost), and ends with a left-to-right message transmission that propagates the LLR in the inverse direction. A PE and its corresponding messages are shown in Fig. 2(a), where $l_{s,i}$ ($r_{s,i}$) denotes the right-to-left (left-to-right) message of the $i$-th node at the $s$-th stage with $i \in \{0, 1, \ldots, N-1\}$ and $s \in \{0, 1, \ldots, n\}$.

The polar BP factor graph can be unrolled [21], [22], and the polar BP iterative decoding can be performed sequentially on the unrolled factor graph. Correspondingly, a PE can be equivalent to two sub-PEs of the unrolled factor graph: one (PEL) updates the right-to-left messages and the other (PER) updates the left-to-right messages, as shown in Fig. 2(b) and Fig. 2(c) respectively, where $t \leq T$ denotes the $t$-th iteration and $T$ is the predetermined maximum iteration number. The two sub-PEs have the same computing structure, but the input and output messages are different.

The update rule of the PEL is

$$\begin{cases} l_{s,i}^t = g(l_{s+1,i}^t, r_{s,i+2^s}^{t-1} + l_{s+1,i+2^s}^t), \\ l_{s,i+2^s}^t = g(l_{s+1,i}^t, r_{s,i}^{t-1}) + l_{s+1,i+2^s}^t, \end{cases} \tag{4}$$

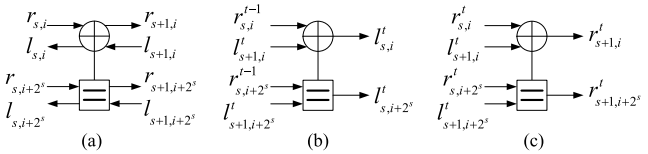



**FIGURE 2.** (a) PE, (b) PEL of the unrolled factor graph and (c) PER of the unrolled factor graph.

and of the PER is

$$\begin{cases} r^t_{s+1,i} = g(r^t_{s,i}, l^t_{s+1,i+2^s} + r^t_{s,i+2^s}), \\ r^t_{s+1,i+2^s} = g(r^t_{s,i}, l^t_{s+1,i}) + r^t_{s,i+2^s}, \end{cases} \tag{5}$$

where, $g(x, y) = \log \frac{1+e^{x+y}}{e^x+e^y}$. In order to reduce the computational complexity, $g(x, y)$ can be approximated as

$$g(x, y) = \text{sgn}(x)\text{sgn}(y)\min(|x|, |y|), \tag{6}$$

where $\text{sgn}(x)$ is the sign function and

$$\text{sgn}(x) = \begin{cases} 1, & x \geq 0, \\ -1, & x < 0. \end{cases} \tag{7}$$

The messages are initialized before decoding. For $i = 0, 1, \ldots, N-1$ and $\forall t \leq T$, the $l^t_{n,i}$ is set to $L_i$ and the $r^t_{0,i}$ is initialized as follows:

$$r^t_{0,i} = \begin{cases} 0, & \text{for } i \in \mathcal{A}, \\ +\infty, & \text{for } i \in \mathcal{A}^c. \end{cases} \tag{8}$$

All the other messages to be propagated are initialized to 0. The polar BP decoder simultaneously estimates the source $\mathbf{u}$ after $T$ iterations such that:

$$\hat{u}_i = \begin{cases} 0, & \text{if } l^t_{0,i} + r^t_{0,i} \geq 0, \\ 1, & \text{otherwise.} \end{cases} \tag{9}$$

## III. SYSTEM MODEL

This section provides the zero-delay system model on the AWGN channel in presence of a one-bit quantizer and introduces the equivalent channel of the AWGN channel after one-bit quantized.

### A. THE SYSTEM MODEL WITH A ONE-BIT QUANTIZER

The system model is an end-to-end communication link equipped with a one-bit quantizer, which is shown in Fig. 3. At the transmitting side, the source vector $\mathbf{u}$ is encoded into a codeword $\mathbf{x}$, and the transmitted signal $\mathbf{s}$ is obtained by BPSK modulation. On the other hand, the receiver is equipped with a one-bit quantizer, which allows the polar decoder to receive only one-bit quantized signal $\mathbf{y}_b$, corresponding to the channel output $\mathbf{y}$. The LLR vector $\mathbf{L}$ that sent to the decoder is calculated by the $\mathbf{y}_b$ instead of $\mathbf{y}$.

The one-bit quantization results in significant performance loss compared to the floating-point polar decoder. In this paper, a neural network decoder (NND) is built to replace the conventional polar decoder. The NND can automatically extract features from the one-bit received signals and learn to correct the transmission errors. Compared with the conventional polar decoder, it shows better error correction performance. The details will be presented below.

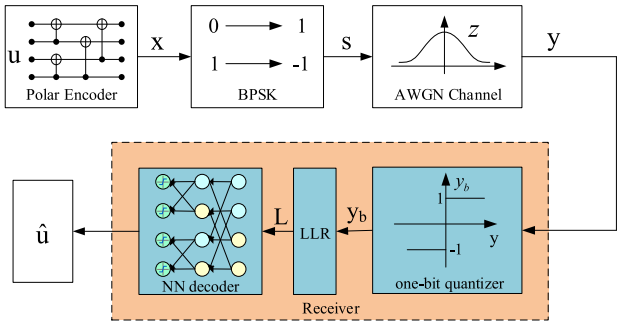



**FIGURE 3.** The zero-delay system model on the AWGN channel in presence of the one-bit quantizer and the neural network decoder.

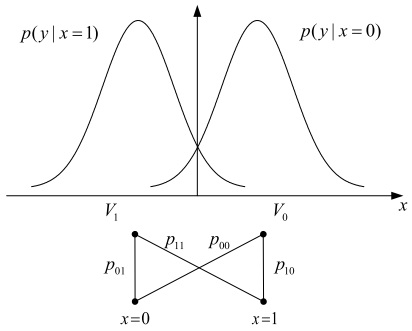



**FIGURE 4.** The equivalent channel of a BIAWGN channel with one-bit quantized output.

### B. EQUIVALENT CHANNEL

In a communication system, signal processing modules such as MIMO signal detection, demodulation and equalization are required before decoding. The signal observed by the decoder is the LLR, which exhibits a Gaussian-like distribution after the previous signal processing. Therefore, the zero-delay system with the AWGN and BPSK are considered in this paper.

The output of the binary input AWGN (BIAWGN) channel is quantized to a binary symbol after passing a one-bit quantizer, and thus can be viewed as a binary symmetric channel (BSC). A demonstration of the equivalent BSC is shown in Fig. 4. The transition probabilities of the equivalent BSC are calculated as follows:

$$P_{ij} = \Pr(y_i \in V_j | x_i = b) = \int_{V_j} p(y_i | x_i = b) \mathrm{d}y_i, \tag{10}$$

where $b, j \in \{0, 1\}$, $x_i$ represents the $i$-th transmitted symbol, and $y_i$ is the corresponding unquantized channel output value. The $V_j$ is an interval with $V_0 = (-\infty, 0)$ and $V_1 = (0, +\infty)$. Given the transmitted value $x_i = b$, the corresponding conditional probability density function (PDF) of the unquantized channel output value $y_i$ is indicated by $p(y_i|x_i = b)$.

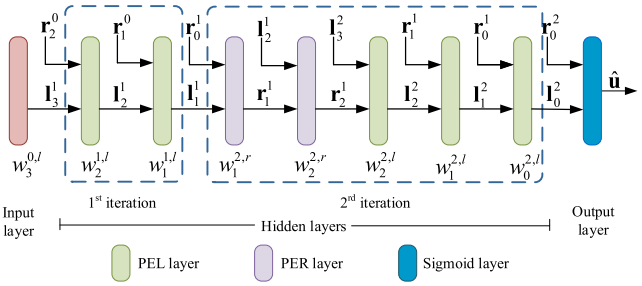



**FIGURE 5.** The structure of NND with $T = 2$ and $N = 8$.

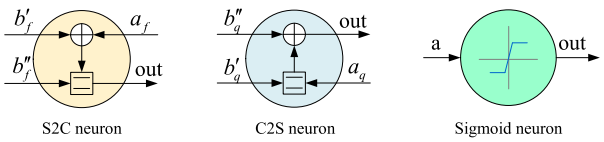



**FIGURE 6.** Three types of neurons in the NND.

## IV. THE NEURAL NETWORK DECODING FRAMEWORK WITH ONE-BIT QUANTIZER

For the low-power practical scenarios considered in this paper, we design an NND to compensate for the performance loss caused by one-bit quantization. In this section, we describe the process of constructing an NND with a layer-based weighting scheme in detail. Besides, we prove that the NND has a symmetrical structure and can be trained with an all-zero codeword.

### A. THE NND STRUCTURE

The NND is constructed based on the unfolded factor graph of polar BP algorithm described in section II-B with a set of active functions according to message update functions.

A simple example representing the NND structure is given in Fig. 5. The NND consists of one input layer, $H = (T - 1) \cdot 2(n - 1) + n$ hidden layers and one output layer, where $n = \log_2(N)$. $\mathbf{l}_s^t$ ($\mathbf{r}_s^t$) denotes the vector of right-to-left (left-to-right) message at the $t$-th iteration and the $s$-th stage of the polar BP factor graph. Especially, $\mathbf{l}_s^0$ ($\mathbf{r}_s^0$) is the value initialized before decoding. Note that the last hidden layer in right-to-left propagation only exists in the last iteration, which calculates the output of the leftmost nodes in the polar BP factor graph.

The NND propagates $l_{s,i}^t$ and $r_{s,i}^t$ that calculated by:

$$\begin{cases} l_{s,i}^t = w_{s,i}^{t,l} \cdot g(l_{s+1,2i-1}^t, l_{s+1,2i}^t + r_{s,i+N/2}^t), \\ l_{s,i+N/2}^t = w_{s,i+N/2}^{t,l} \cdot g(l_{s+1,2i-1}^t, r_{s,i}^t) + l_{s+1,2i}^t, \\ r_{s+1,2i-1}^t = w_{s+1,2i-1}^{t,r} \cdot g(r_{s,i}^t, l_{s+1,2i}^{t-1} + r_{s,i+N/2}^t), \\ r_{s+1,2i}^t = w_{s+1,2i}^{t,r} \cdot g(r_{s,i}^t, l_{s+1,2i-1}^{t-1}) + r_{s,i+N/2}^t, \end{cases} \tag{11}$$

where $w_{s,i}^{t,l}$ ($w_{s,i}^{t,r}$) denotes the trainable weight of the left (right) message that assigned to the $i$-th neuron at $s$-th stage of $t$-th iteration. There are three types of neurons in the NND to compute the propagating messages, as shown in Fig. 6, where a S2C (sum to check) neuron and a C2S (check to sum) neuron constitute a sub-PE (PEL or PER), and S2C neuron is used to calculates $l_{s,i}^t$ and $r_{s+1,2i-1}^t$, while C2S neuron calculates $l_{s,i+N/2}^t$ and $r_{s+1,2i}^t$. The active functions of S2C neuron and C2S neuron are defined as follows:

$$f(a_f, b_f', b_f'', w_f) = w_f \cdot g(b_f'', a_f + b_f'), \tag{12}$$

$$q(a_q, b_q', b_q'', w_q) = w_q \cdot g(a_q, b_q') + b_q'', \tag{13}$$

where $a_f$, $b_f'$ and $b_f''$ are the input messages to S2C neuron, and $b_f'$ and $b_f''$ belong to the same vector $\mathbf{b}_f$, i.e., $\mathbf{b}_f = \{b_f', b_f''\}$. Similarly, $a_q$, $b_q'$ and $b_q''$ are the input messages to C2S neuron, and $\mathbf{b}_q = \{b_q', b_q''\}$. $w_f$ and $w_q$ are the trainable weights, related to the location of the neurons in the NND.

In the output layer, the active function of the sigmoid neuron is expressed as:

$$\sigma(a) = \frac{1}{1 + \mathrm{e}^{-a}}. \tag{14}$$

### B. LAYER-BASED WEIGHTING SCHEME

Neural networks optimize performance by constantly adjusting weights during training. Especially for model-driven neural network decoders, designing weighting schemes based on conventional communication models is the most important task. In the conventional BP-NNs, each neuron is assigned an independent weight. However, too many weights will lead to the deviation of the learning direction and hinder the learning speed.

In this paper, a layer-based weighting scheme is proposed. We impose the constraints on the weights to guide the learning process. The layer-based weighting scheme is defined in (15), where $w_s^{t,l}$ ($w_s^{t,r}$) denotes the trainable weight of the left (right) message that assigned to the $s$-th layer of $t$-th iteration. Each layer of the NND shares a common weight and the reduction of trainable parameters leads to a fast training convergence.

$$\begin{cases} l_{s,i}^t = w_s^{t,l} \cdot g(l_{s+1,2i-1}^t, l_{s+1,2i}^t + r_{s,i+N/2}^t), \\ l_{s,i+N/2}^t = w_s^{t,l} \cdot g(l_{s+1,2i-1}^t, r_{s,i}^t) + l_{s+1,2i}^t, \\ r_{s+1,2i-1}^t = w_{s+1}^{t,r} \cdot g(r_{s,i}^t, l_{s+1,2i}^{t-1} + r_{s,i+N/2}^t), \\ r_{s+1,2i}^t = w_{s+1}^{t,r} \cdot g(r_{s,i}^t, l_{s+1,2i-1}^{t-1}) + r_{s,i+N/2}^t. \end{cases} \tag{15}$$

## V. SYMMETRY OF THE NND

Symmetry is crucial for training the NND, which determines whether NND can be trained with an all-zero codeword. Inspired by the symmetry property of the sum product decoding algorithm [26], we prove that the proposed NND also has a symmetrical structure under the influence of multiple unequal weights.

For the convenience of explanation, we give an equivalent definition of the active function of the S2C neuron. Let $m_f = a_f + b_f''$, then the active function of the S2C neuron can be expressed as

$$f(m_f, b_f', w_f) = w_f \cdot g(m_f, b_f'), \tag{16}$$

where $m_f$ and $b_f'$ can be considered as two LLR messages propagated on the NND.

*Lemma 1 (Symmetry of S2C Neuron): Given two independent binary random variables $a_1$ and $a_2$ with probabilities $\Pr(a_i = b) = p_b^{(i)}$, $b \in \{0, 1\}$, and $L_i = L(a_i) = \ln(p_0^{(i)}/p_1^{(i)})$. if*

$$f(L_1, L_2, w_f) = \mathrm{sgn}(L_1)\mathrm{sgn}(L_2) \cdot f(|L_1|, |L_2|, w_f), \quad (17)$$

*then the S2C neuron is symmetrical, i.e. the error probability is independent of the transmitted vector.*

*Proof:* The LLR of the binary sum $A_{1,2} = a_1 \oplus a_2$ is calculated as follows:

$$\begin{aligned} L(A_2) &= \ln\left(\frac{\Pr(A_{1,2}=0)}{\Pr(A_{1,2}=1)}\right) = \ln\left(\frac{\Pr(a_1 \oplus a_2 = 0)}{\Pr(a_1 \oplus a_2 = 1)}\right) \\ &= \ln\left(\frac{p_0^{(1)}p_0^{(2)} + p_1^{(1)}p_1^{(2)}}{p_0^{(1)}p_1^{(2)} + p_1^{(1)}p_0^{(2)}}\right) = \ln\left(\frac{1 + \frac{p_0^{(1)}}{p_1^{(1)}}\frac{p_0^{(2)}}{p_1^{(2)}}}{\frac{p_0^{(1)}}{p_1^{(1)}} + \frac{p_0^{(2)}}{p_1^{(2)}}}\right) \\ &= \ln\left(\frac{1 + e^{L_1+L_2}}{e^{L_1} + e^{L_2}}\right). \end{aligned} \quad (18)$$

This result can be directly applied to the computation of S2C neuron. For the input LLR messages $m_f = a_f + b_f''$ and $b_f'$, defined in (16), the $g$ operation is performed as

$$\begin{aligned} g(m_f, b_f') &= \ln\left(\frac{1 + e^{m_f + b_f'}}{e^{m_f} + e^{b_f'}}\right) \\ &= 2\tanh^{-1}\left(\tanh\left(\frac{m_f}{2}\right)\tanh\left(\frac{b_f'}{2}\right)\right). \end{aligned} \quad (19)$$

where, $\tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$ and $\tanh^{-1}(x)$ is the inverse function of the $\tanh(x)$. Let $\alpha = \mathrm{sgn}(m_f)$ and $\beta = \mathrm{sgn}(b_f')$. We have

$$g(m_f, b_f') = \alpha\beta \cdot 2\tanh^{-1}\left(\tanh\left(\frac{|m_f|}{2}\right)\tanh\left(\frac{|b_f'|}{2}\right)\right), \quad (20)$$

for any combination $\{m_f, b_f'\}$. so the $g$ operation is symmetrical.

Introduce the weight $w_f$, the output of the S2C neuron ($d_f$) is calculated as follows:

$$\begin{aligned} d_f &= f(m_f, b_f', w_f) \\ &= \alpha\beta \cdot w_f \cdot 2\tanh^{-1}\left(\tanh\left(\frac{|m_f|}{2}\right)\tanh\left(\frac{|b_f'|}{2}\right)\right). \end{aligned} \quad (21)$$

Therefore, S2C neuron is symmetrical. □

*Lemma 2 (Symmetry of C2S Neuron): The activation function of C2S neuron satisfies the invariance of sign inversion, i.e.,*

$$q(-a_q, -b_q', -b_q'', w_q) = -q(a_q, b_q', b_q'', w_q), \quad (22)$$

*where, $a_q$, $b_q'$ and $b_q''$ are the inputs of the C2S neuron, as defined in (13).*

*Proof:* Based on the similar priciples, the output of the C2S neuron ($d_q$) is calculated by:

$$\begin{aligned} d_q &= q(-a_q, -b_q', -b_q'', w_q) \\ &= w_q \cdot g(-a_q, -b_q') + (-b_q'') \\ &= w_q \cdot (-(g(a_q, b_q') + b_q'')). \end{aligned} \quad (23)$$

Hence, we can see that the C2S neuron also has a symmetrical structure. □

The symmetry of the S2C neuron and C2S neuron can be generalized to the entire neural network decoder.

*Theorem 1 (Symmetry of the NND): Let $\mathbf{d}_0$ denote the output vector of the input layer and $\mathbf{d}_h$ denote the output vector of the h-th hidden layer. Suppose $\Gamma(\cdot)$ present the propagation function of the h-th hidden layer, this function can be recursively expressed as*

$$\Gamma(d_0) = \Phi_h(\Phi_{h-1}(\ldots\Phi_1(\mathbf{d}_0, w_1), \ldots, w_{h-1}), w_h), \quad (24)$$

*where, for $\forall h \in \{1, 2, \ldots, H\}$ and $i = 0, 1, \ldots, N-1$, $w_h$ is the trainable weight assigned to hidden layer, $\Phi_h = \{\varphi_{h,i}\}$ is a collection of active functions, and*

$$\varphi_{h,i} = \begin{cases} f(a_f, b_f', b_f'', w_f), & \text{if S2C neuron,} \\ q(a_q, b_q', b_q'', w_q), & \text{if C2S neuron.} \end{cases} \quad (25)$$

*Thus, the NND satisfies a symmetrical structure, that is*

$$\mathbf{d}_h = -\Gamma(-\mathbf{d}_0), \quad (26)$$

*Proof:* Recursive propagation of LLRs through two types neuron (S2C and C2S) can be decomposed into single-step transformations of LLRs.

Lemma 1 and Lemma 2 have given the symmetry of the NND with $N = 2$:

$$\mathbf{d}_1 = -\Phi_1(-\mathbf{d}_0, w_1). \quad (27)$$

For any positive integer $n$, we assume that the NND with $N = 2^{n-1}$ has the symmetrical structure

$$\mathbf{d}_{h-1} = -\Phi_{h-1}(\cdots - \Phi_1(-\mathbf{d}_0, w_1), \ldots, w_{h-1}). \quad (28)$$

The output of the $h$-th hidden layer in the NND with $N = 2^n$ are calculated as follows:

$$\mathbf{d}_h = -\Phi_h(-\mathbf{d}_{h-1}) = -\Gamma(-\mathbf{d}_0) \quad (29)$$

It is mathematically proved that the NND has a symmetrical structure on the basis of Lemma 1 and Lemma 2. Furthermore, the symmetrical structure exists in all BP-NNs. □

With the symmetry condition on the neural network decoder and the equivalent channel, the whole behaviors of the neural network decoder can be predicted from the behavior of assuming that an all-zero codeword is transmitted. In other words, an all-zero codeword is enough to train the BP-NNs.

## VI. EXPERIMENTS AND NUMERICAL RESULTS

In this section, the proposed NND is implemented. We verify the performance of the NND with a one-bit quantizer in different codeword configurations. In addition, we provide a performance comparison between the NND and several typical previous works.

### A. TRAINING THE NND

Based on the symmetrical structure of the NND, the training dataset is generated by transmitting an all-zero codeword. The codeword $\mathbf{x}$ is modulated by BPSK modulation and transmitted over an AWGN channel with mean of 0 and variance of $\sigma^2$. Thus, after one-bit quantization, the receive signal vector can be expressed as $\mathbf{y}_b = \text{sgn}(\mathbf{1}^N + \mathbf{z})$, where $\mathbf{1}^N$ represents an all-one vector with length $N$, which is obtained by modulation of the all-zero codeword $\mathbf{x}$. In this paper, we use the LLR vector as the input to the NND,

$$\mathbf{L} = \frac{2\text{sgn}(\mathbf{1}^N + \mathbf{z})}{\sigma^2}. \tag{30}$$

It should be clarified that the LLR in this paper cannot accurately reflect the probability that the source bit is 0 or 1, due to the use of the one-bit quantizer. The LLR calculated by (30) can be regarded as a softened hard decision result to facilitate the learning of the neural network.

There are several candidates for loss function as listed in [19], and we consider the following cross-entropy loss function:

$$E = -\frac{1}{N}\sum_{i=1}^{N} u_i \log(o_i) + (1-u_i)\log(1-o_i), \tag{31}$$

where $\mathbf{u}$ is the source vector and $\mathbf{o} = \{o_0, o_1, \ldots, o_{N-1}\}$ is the output vector of the NND.

We use supervised learning and mini-batch stochastic gradient descent (SGD) to train the NND. The gradient is calculated by the back-propagation algorithm [24] efficiently. The model is implemented by the deep learning framework Tensorflow [25].

### B. DECODING EXPERIMENTS OF (64,32) POLAR CODES

In this case, we consider the typical (64,32) polar codes which are constructed using Gaussian approximation (GA) [27]. Several parameters are pre-set before the experiments. The size of the batch for training and testing is set to 140 with the signal to noise ratio (SNR) varying SNRs = {3,3.5,4,4.5,5,5.5,6}dB and 20 samples per SNR. The size of the training dataset $S$ is set to $1 \times 10^4$ batches. To ensure adequate training samples, $S$ is set to a large value. However, so many training samples may not be necessary in actual training. We test the block error rate (BLER) performance of the NND after every 100 batches. It should be emphasized that the training dataset is generated by an all-zero codeword, but the testing dataset is a collection of random codewords.

The NND proposed in this paper compares the BLER performance with the conventional polar BP algorithm [1] and the BP-NNs proposed in [18], [21] and [23], respectively. The notation "$A$-$T$-$Q$bit" describes that the "$A$" algorithm performs "$T$" iterations and "$Q$ bit" quantized signal vector is received.

The BLER performance of (64,32) polar codes is shown in Fig. 7. Under the constraint of 5 iterations, the proposed NND with a one-bit quantizer has a 1.2dB gain compared to the BP-5-1bit, and 0.2dB gain compared to BP-5-2bit. Moreover, the NND-5-1bit has a lower BLER curve than the BP-30-1bit. We also compare the performance between the layer-based weighting scheme (NND-5-1bit) and the conventional neuron-based weighting scheme (MSBP-5-1bit). It can be observed that the layer-based weighting scheme results in a small performance gain at high SNR. Therefore, the decoding performance of BP-NNs is affected by weights distribution and too many weights will cause performance loss. Moreover, the BLER results of the SNND-5-1bit [23] and the Neural BP-5-1bit [18] are tested. Observation shows that the proposed NND has the lowest BLER curve compared to them.

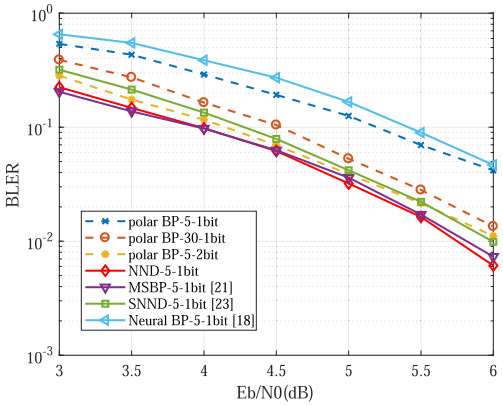



**FIGURE 7.** The BLER performance of the NND with (64, 32) polar codes.

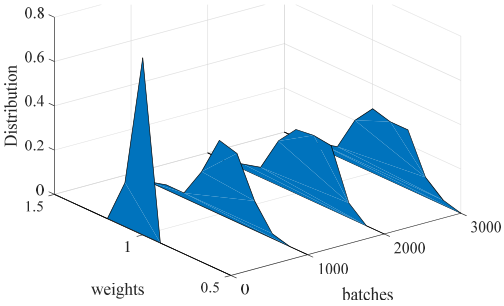



**FIGURE 8.** The probability distribution of the weights in the NND, which is build based on the unfolded (64,32) polar factor graph with 5 iterations.

Fig. 8 shows the distribution of weights in the NND of (64,32) polar codes. The distribution approximates a Gaussian distribution instead of a fixed value of 1. This means that the trained weights change the LLRs in the standard polar BP algorithm. At the beginning of the training, the distribution has a small variance, and as the training progresses, it becomes wider until stable. This indicates that the training is effective, and the training process converges by the steady distribution.

### C. DECODING EXPERIMENTS OF (256,128) POLAR CODES

In this case, the BLER performance of (256,128) polar codes is tested. We retain the same pre-set parameters and the same SNR range as the experiments of (64,32) polar codes. The BLER of (256,128) polar codes is shown in Fig. 9. We can see from the figure that the NND-5-1bit achieves a

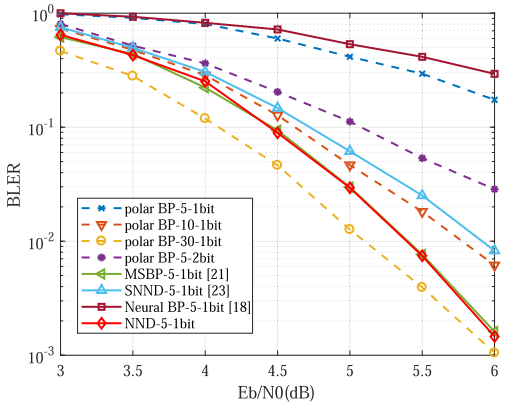



**FIGURE 9.** The BLER performance of the NND with (256,128) polar codes.

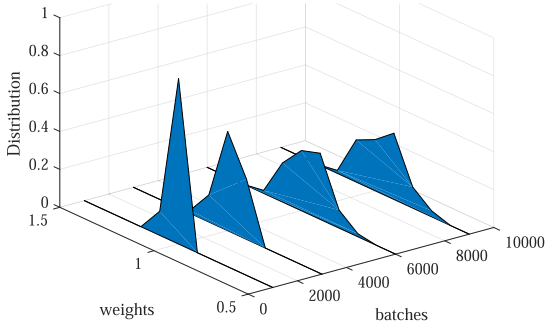



**FIGURE 10.** The probability distribution of the weights in the NND, which is build based on the unfolded (256,128) polar factor graph with 5 iterations.

2.0dB gain compared to the BP-5-1bit. In addition, the BLER performance of NND-5-1bit is better than the BP-10-1bit, BP-5-2bit, SNND-5-1bit [23] and Neural BP-5-1bit [18]. For the (256,128) polar codes, the layer-based weighting scheme (NND-5-1bit) also result in performance gain compared with the conventional neuron-based weighting scheme (MSBP-5-1bit). However, there is a gap between the NND-5-1bit and the BP-30-1bit, but the gap becomes smaller as the SNR increases.

The distributions of the weights in the NND of (256, 128) polar codes are shown in Fig. 10. It is similar to (64,32) polar codes. However, compared to the NND of (64,32) polar codes, the NND of (256,128) polar codes requires more training batches to achieve convergence due to more training weights.

### D. VERIFY CONVERGENCE

The convergence of the proposed NND and standard polar BP algorithm with one-bit quantizer is verified and the results are shown in Fig. 11. We can see that the BLER curve of the NND has a larger slope, compared with the standard polar BP algorithm. Therefore, with the aid of trained weights, the proposed NND converges faster than the standard polar BP algorithm. Furthermore, the proposed NND has a lower BLER curve than the standard polar BP algorithm, which proves that the trained weights also obtain performance gain.

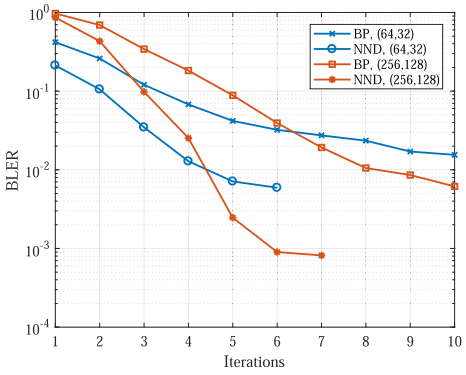



**FIGURE 11.** The comparison of convergence between the NND and the conventional BP algorithm with one-bit quantizer, where $N = 64, 256$ and Eb/N0 = 6dB.

**TABLE 1.** The number of training batches that achieve the BLER performance of this paper.

| codeword | (64,32) | | (256,128) | |
|---|---|---|---|---|
| model | MSBP | NND | MSBP | NND |
| batches | $4.9 \times 10^3$ | $3.0 \times 10^3$ | $1.3 \times 10^4$ | $8.5 \times 10^3$ |

### E. LEARNING EFFICIENCY

In general, more trainable parameters often mean more training samples are needed. In this paper, our layer-based weighting scheme limits the number of trainable weights, and caused the neural network to converge fast. Table 1 lists the number of training batches that achieve the desired BLER performance as shown in Fig. 7 and Fig. 9. We can see from the table that the NND achieves the BLER performance with fewer training batches than MSBP, which means that the proposed layer-based weighting scheme improves the learning efficiency of the neural network.

## VII. CONCLUSION

In this paper, we apply deep learning technology to decode polar codes with a one-bit quantizer and proposed a neural network decoder. Our decoder is built based on the unfolded polar BP factor graph. The symmetrical structure of the neural network decoder is proved in the paper, which allows the neural network decoder can be trained with an all-zero codeword and noise realization only. A layer-based weighting scheme is proposed, which results in faster learning than the conventional BP-NNs. The neural network decoder compensates for the performance loss caused by the one-bit quantizer and a 2dB gain can be obtained in the decoding of polar codes with medium code length.

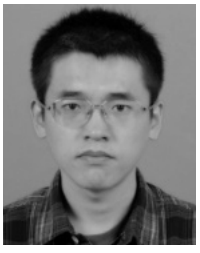
## REFERENCES

[1] E. Arikan, "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, Jul. 2009.
[2] I. Tal and A. Vardy, "List decoding of polar codes," *IEEE Trans. Inf. Theory*, vol. 61, no. 5, pp. 2213–2226, May 2015.
[3] K. Niu and K. Chen, "CRC-aided decoding of polar codes," *IEEE Commun. Lett.*, vol. 16, no. 10, pp. 1668–1671, Oct. 2012.

[4] K. Chen, K. Niu, and J. Lin, "Improved successive cancellation decoding of polar codes," *IEEE Trans. Commun.*, vol. 61, no. 8, pp. 3100–3107, Aug. 2013.
[5] *Multiplexing and Channel Coding*, document 3GPP TS 38.212, Release 15, V15.2.0, 2018.
[6] O. Orhan, E. Erkip, and S. Rangan, "Low power analog-to-digital conversion in millimeter wave systems: Impact of resolution and bandwidth on performance," in *Proc. Inf. Theory Appl. Workshop (ITA)*, San Diego, CA, USA, Feb. 2015, pp. 191–198.
[7] Y.-S. Jeon, H. Do, S.-N. Hong, and N. Lee, "Soft-output detection methods for sparse millimeter-wave MIMO systems with low-precision ADCs," *IEEE Trans. Commun.*, vol. 67, no. 4, pp. 2822–2836, Apr. 2019.
[8] E. Balevi and J. G. Andrews, "One-bit OFDM receivers via deep learning," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4326–4336, Jun. 2019.
[9] Y.-S. Jeon, N. Lee, and H. V. Poor, "Robust data detection for MIMO systems with one-bit ADCs: A reinforcement learning approach," 2019, *arXiv:1903.12546*. [Online]. Available: https://arxiv.org/abs/1903.12546
[10] G. Fettweis and S. Alamouti, "5G: Personal mobile internet beyond what cellular did to telephony," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 140–145, Feb. 2014.
[11] M. Varasteh, O. Simeone, and D. Gunduz, "Joint source-channel coding with one-bit ADC front end," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 3062–3066.
[12] M. Liu, T. Song, J. Hu, J. Yang, and G. Gui, "Deep learning-inspired message passing algorithm for efficient resource allocation in cognitive radio networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 641–653, Jan. 2019.
[13] Y. Wang, M. Liu, J. Yang, and G. Gui, "Data-driven deep learning for automatic modulation recognition in cognitive radios," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 4074–4077, Apr. 2019.
[14] J. Sun, W. Shi, Z. Yang, J. Yang, and G. Gui, "Behavioral modeling and linearization of wideband RF power amplifiers using BiLSTM networks for 5G wireless systems," *IEEE Trans. Veh. Technol.*, vol. 68, no. 11, pp. 10348–10356, Nov. 2019.
[15] H. Huang, Y. Peng, J. Yang, W. Xia, and G. Gui, "Fast beamforming design via deep learning," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 1065–1069, Jan. 2020, doi: 10.1109/tvt.2019.2949122.
[16] T. Gruber, S. Cammerer, J. Hoydis, and S. T. Brink, "On deep learning-based channel decoding," in *Proc. 51st Annu. Conf. Inf. Sci. Syst. (CISS)*, Baltimore, MD, USA, Mar. 2017, pp. 1–6.
[17] X. Liu, S. Wu, Y. Wang, N. Zhang, J. Jiao, and Q. Zhang, "Exploiting error-correction-CRC for polar SCL decoding: A deep learning based approach," *IEEE Trans. Cogn. Commun. Netw.*, to be published, doi: 10.1109/tccn.2019.2946358.
[18] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *Proc. 54th Annu. Allerton Conf. Commun., Control, Comput. (Allerton)*, Monticello, IL, USA, Sep. 2016, pp. 341–346.
[19] 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 J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 119–131, Feb. 2018.
[20] L. Lugosch and W. J. Gross, "Neural offset min-sum decoding," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Aachen, Germany, Jun. 2017, pp. 1361–1365.
[21] W. Xu, Z. Wu, Y.-L. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Lorient, France, Oct. 2017, pp. 1–6.
[22] N. Doan, S. A. Hashemi, E. N. Mambou, T. Tonnellier, and W. J. Gross, "Neural belief propagation decoding of CRC-polar concatenated codes," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.
[23] W. Xu, X. You, C. Zhang, and Y. Berery, "Polar decoding on sparse graphs with deep learning," in *Proc. 52nd Asilomar Conf. Signals, Syst., Comput.*, Pacific Grove, CA, USA, Oct. 2018, pp. 599–603.
[24] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: http://www.deeplearningbook.org
[25] M. Abadi, A. Agarwal, and P. Barham, "TensorFlow: Large-scale machine learning on heterogeneous distributed systems," 2016, *arXiv:1603.04467*. [Online]. Available: https://arxiv.org/abs/1603.04467
[26] T. Richardson and R. Urbanke, "The capacity of low-density parity-check codes under message-passing decoding," *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 599–618, Feb. 2001.
[27] P. Trifonov, "Efficient design and decoding of polar codes," *IEEE Trans. Commun.*, vol. 60, no. 11, pp. 3221–3227, Nov. 2012.

**JIAN GAO** (Student Member, IEEE) received the B.S. degree in electronical information science and technology from the Shandong University of Science and Technology, Qingdao, China, in 2014. He is currently pursuing the Ph.D. degree with the School of Information and Communication Engineering, Beijing University of Posts and Telecommunications. His research interests include information theory, coding theory, and deep learning.

**KAI NIU** (Member, IEEE) received the B.S. degree in information engineering and the Ph.D. degree in signal and information processing from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 1998 and 2003, respectively. He is currently a Professor with the School of Information and Communication Engineering, BUPT. His research interests include coding theory and its applications, space-time codes, and broadband wireless communication. Since 2008, he has been a Senior Member with the Chinese Institute of Electronics and a Committee Member of the Information Theory Chapter of CIE. He is also an Associate Editor with the IEEE COMMUNICATIONS LETTERS.



**CHAO DONG** (Member, IEEE) received the B.S. and Ph.D. degrees in signal and information processing from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2007 and 2012, respectively. He is currently an Associate Professor with the School of Information and Communication Engineering, BUPT. His research interests include the MIMO signal processing, multiuser precoding, decision feedback equalizer, and relay signal processing.