

doi:10.13756/j.gtxyj.2022.04.004

光通信系统与网络技术

改进的循环神经网络极化码 BP 译码算法

邓学璐, 彭大芹

(重庆邮电大学, 通信与信息工程学院, 重庆 400065)

摘要:近年来,新兴的深度学习(DL)技术在译码领域取得了进展,目前提出的极化码神经网络译码器有着比置信传播(BP)译码更快的收敛速度和更好的误码率(BER)性能,但其仍存在运算复杂度高的问题。文章针对该问题,在迭代过程中采用改进信息更新这个思路,提出了一种改进左信息更新的循环神经网络(RNN)偏移最小和(OMS)近似 BP(RNN-OMS-BP-L)译码算法。仿真结果表明,文章所提算法相比于深度神经网络(DNN)BP(DNN-BP)译码算法,使用 6.25% 的加法运算代价替换了全部的乘法运算;相比于目前较优的 RNN OMS 近似 BP(RNN-OMS-BP)译码算法,在确保 BER 性能几乎无损失的前提下,文章所提算法使用改进的信息更新减少了 25% 的加法运算复杂度,节省了部分存储空间开销,在相同 BER 性能下,减少了 37.5% 的迭代次数。

关键词:极化码;置信传播;循环神经网络;信息更新;低运算复杂度

中图分类号:TN929.5

文献标志码:A

文章编号:1005-8788(2022)04-0017-06

Improved Recurrent Neural Network based BP Decoding Algorithm for Polar Codes

DENG Xue-lu, PENG Da-qin

(School of Communication and Information Engineering, Chongqing University
of Posts and Telecommunications, Chongqing 400065, China)

Abstract: In recent years, the emerging Deep Learning (DL) technology has made progress in the field of decoding. Current polar code neural network decoder has faster convergence speed and better Bit Error Rate (BER) performance than Belief Propagation (BP) decoding. However, it still has the problem of high computational complexity. Therefore, in order to improve this problem, this paper adopts the idea of improving information update in the iterative process, and proposes a Recurrent Neural Network (RNN) Offset Min-Sum (OMS) BP decoding algorithm that improves Left information update (RNN-OMS-BP-L). The simulation results show that, compared with the Deep Neural Network (DNN) BP (DNN-BP) decoding algorithm, this algorithm replaces all multiplications with a cost of 6.25% addition. Compared with the current optimized RNN OMS and approximate BP (RNN-OMS-BP) decoding algorithm, the decoding algorithm in this paper uses improved information to reduce 25% of the addition operations with almost no loss in BER performance, while saving part of the storage space overhead. Under the same BER performance, it reduces the number of iterations by 37.5%.

Key words: polar code; BP; RNN; information updates; low complexity

0 引 言

极化码是最开始由 Arikan 在文献[1]中提出的一种能够实现接近信道容量的分组码,是第五代移动通信技术(5th Generation Mobile Communication Technology, 5G)信道编码方案。串行抵消(Successive Cancellation, SC)和置信传播(Belief Propagation, BP)译码是极化码中较普遍的两种译码算法。BP 译码因其并行译码过程,比 SC^[2-3]译码具有更高的吞吐量和更低的译码时延,但采用偏移最小和(Offset Min-Sum, OMS)近似^[4]与早期停止准则^[5]来降低冗余迭代次数和运算复杂度的 BP 译码算法在误码率(Bit Error Ratio, BER)性能上不如 SC 译码算法。

文献[6-9]中,BP 译码性能优化的一种方法是引入神经网络辅助 BP 译码过程。但上述文献中提出的方法大部分随着极化码码长和迭代次数的增加,左右信息更新运算复杂度也大量增加,这阻碍了神经网络在实际通信系统中的应用。因此,针对以上神经网络译码器存在的不足,受文献[8]针对循环神经网络(Recurrent Neural Network, RNN)BP 译码算法的优化及文献[10]针对极化码软信息连续消除(Soft Successive Cancellation, SCAN)译码算法所采用的 simplified-SCAN 算法的启发,本文提出了一种改进左信息更新的 RNN OMS 近似 BP(RNN-OMS-BP-Left, RNN-OMS-BP-L)译码算法,采用新的左信息更新方式,减少一部分节点更新时的计算量和存储开销,并替换初始的 RNN 译码

收稿日期:2021-12-01

基金项目:国家自然科学基金资助项目(E020B2018023)

作者简介:邓学璐(1997—),男,重庆人。硕士,主要研究方向为极化码译码。

通信作者:彭大芹,教授级高级工程师。E-mail:pengdq@cqupt.edu.cn

器,降低了BP算法的运算复杂度和内存占用,减少了迭代次数。

1 极化码编码与BP译码

1.1 极化码

极化码的核心构造原理是“信道极化”,通过信道极化,可以将一组独立的二元输入信道变换成传输信息比特的无噪声信道或固定收发冻结比特的全噪声信道。信道极化包括两个过程:信道合并与信道分裂,如图1所示,其中, W 为二进制离散无记忆信道; W^N 为信道 W 的 N 次使用,即 N 个独立且相同的 W 组成的矢量信道; $W_N^{(i)} (1 \leq i \leq N)$ 为 W 经过信道合并与分裂形成的 N 个新的比特信道。

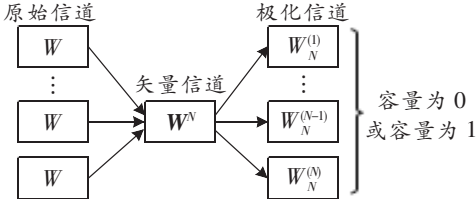


图1 极化码信道合并与分裂

极化码编码原理为:把 $N=2^n$ (n 为任意正整数)极化码中的 K 位信息比特放置在信息比特索引集合 A 的比特信道中, $N-K$ 位冻结比特则放置于集合 A^c 索引中,组成输入信息序列 $u^N = \{u_A, u_{A^c}\}$,式中: u_A 为信息比特; u_{A^c} 为冻结比特。所以码率为 K/N 的 (N, K) 极化码 x^N 可通过生成矩阵 G_N 得到:

$$x^N = u^N G_N = u^N F^{\otimes n} B_N, \quad (1)$$

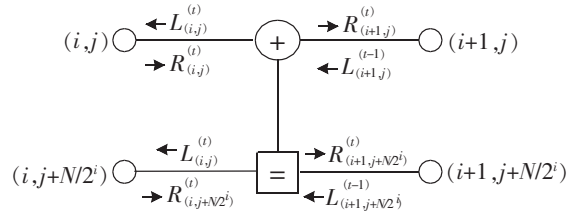
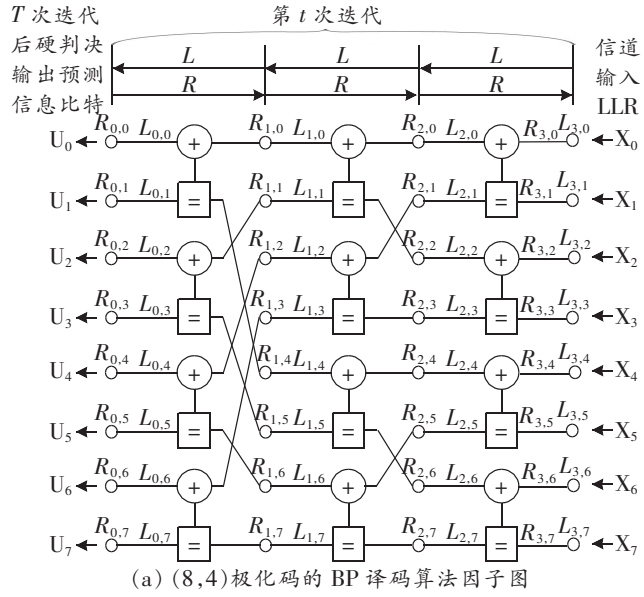
式中: $n = \log_2 N$; $F^{\otimes n}$ 为对 $F = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$ 的次克朗内克积; B_N 为比特反序矩阵。原始比特序列 u^N 通过与生成矩阵 G_N 相乘即可得到长度为 N 的编码比特序列 x^N 。

1.2 极化码的BP译码算法

Arıcan在文献[11]中提出,BP译码也可作为极化码译码算法,并且可采用类似SC译码算法的连续消除规则[12]。极化码的编码结构可直接转换成因子图。由 $n = \log_2 N$ 个阶段及 $N \times (n+1)$ 个节点组成一个 $(8, 4)$ 极化码的BP译码因子图如图2所示。

由图2(b)可得:

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



注: $U_i (0 \leq i < N, N$ 为码长)为信源序列; X_i 为码长序列; i 和 j 分别为图的层数和该节点在本层的序号; $R_{ij}^{(t)}$ 为从左至右传递的右信息对数似然比(Like Likelihood Rate, LLR); $L_{ij}^{(t)}$ 为从右至左传递的左信息LLR。

图2 极化码BP译码算法因子图

式中,

$$g(a, b) = \ln \frac{1+ab}{a+b} \approx \text{sign}(a) \times \text{sign}(b) \min(|a|, |b|). \quad (3)$$

迭代译码前,首先将 $R_{1,j}^{(1)}$ 和 $L_{n+1,j}^{(1)}$ LLR信息初始化。若 u_j 为信源比特,则 $R_{1,j}^{(1)}$ 为信源比特的LLR值,对于冻结比特,将其相应的 $R_{1,j}^{(1)}$ 赋值为已知冻结比特的LLR值, $L_{n+1,j}^{(1)}$ 则赋值为信道接收信息比特的LLR值即可,如下所示:

$$R_{1,j}^{(1)} = \begin{cases} 0, & \text{if } j \in A \\ \infty, & \text{if } j \in A^c \text{ 且 } u_j = 0 \\ -\infty, & \text{if } j \in A^c \text{ 且 } u_j = 1 \end{cases}, \quad (4)$$

$$L_{n+1,j}^{(1)} = \ln \frac{p(y_j | x_j = 0)}{p(y_j | x_j = 1)}, \quad (5)$$

式中: $p()$ 为发生概率; x_j 和 y_j 分别为从信道中发送和接收到的第 j 位信息比特。达到预设的 T 次迭代更新后,通过对最后一次迭代输出的左、右信息LLR $L_{0,j}^{(T)}$ 、 $R_{0,j}^{(T)}$ 进行比特判决得到译码结果:

$$u_j^N = \begin{cases} 0, & \text{if } L_{0,j}^{(T)} + R_{0,j}^{(T)} \geq 0 \\ 1, & \text{if } L_{0,j}^{(T)} + R_{0,j}^{(T)} < 0 \end{cases}. \quad (6)$$

2 神经网络 BP 译码算法

文献[6]提出了基于深度神经网络(Deep Neural Network, DNN)的 BP(DNN-BP)译码算法,在译码迭代过程中的消息更新因子图中设置了可学习的参数,即把式(2)替换为

$$\begin{cases} L_{i,j}^{(t)} = Lv_{i,j}^{(t)} \cdot g(L_{i+1,j}^{(t-1)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}) \\ L_{i,j+N/2^i}^{(t)} = Lv_{i,j+N/2^i}^{(t)} \cdot g(R_{i,j}^{(t)}, L_{i+1,j}^{(t-1)} + L_{i+1,j+N/2^i}^{(t-1)}) \\ R_{i+1,j}^{(t)} = Rv_{i+1,j}^{(t)} \cdot g(R_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}) \\ R_{i+1,j+N/2^i}^{(t)} = Rv_{i+1,j+N/2^i}^{(t)} \cdot g(R_{i,j}^{(t)}, L_{i+1,j}^{(t-1)} + R_{i,j+N/2^i}^{(t)}) \end{cases}, \quad (7)$$

式中, $Lv_{i,j}^{(t)}$ 和 $Rv_{i+1,j}^{(t)}$ 分别为在第 t 次迭代中,第 i 层,序号为 j 的节点从右至左传递的左信息和从左至右传递的右信息的缩放因子。

文献[8]提出了基于 RNN 权重量化机制的 OMS 近似 BP(RNN-OMS-BP)译码算法用于极化码译码。受此启发,本文所提算法使用 OMS 近似算法^[13]替代缩放因子最小和近似算法,式(3)被替换为

$$g'(a, b, \beta) = \text{sign}(a) \text{sign}(b) \times \max(\min(|a|, |b|) - \beta, 0), \quad (8)$$

式中, β 为偏移量。

式(7)迭代更新变为

$$\begin{cases} L_{i,j}^{(t)} = g'(L_{i+1,j}^{(t-1)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}, \beta_{L_{i,j}^{(t)}}) \\ L_{i,j+N/2^i}^{(t)} = g'(R_{i,j}^{(t)}, L_{i+1,j}^{(t-1)} + \beta_{L_{i,j+N/2^i}^{(t)}}) + L_{i+1,j+N/2^i}^{(t-1)} \\ R_{i+1,j}^{(t)} = g'(R_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}, \beta_{R_{i+1,j}^{(t)}}) \\ R_{i+1,j+N/2^i}^{(t)} = g'(R_{i,j}^{(t)}, L_{i+1,j}^{(t-1)} + \beta_{R_{i+1,j+N/2^i}^{(t)}}) + R_{i+1,j+N/2^i}^{(t-1)} \end{cases}, \quad (9)$$

式中, $\beta_{L_{i,j}^{(t)}}$ 和 $\beta_{R_{i+1,j}^{(t)}}$ 分别为第 t 次迭代更新译码过程中进行左信息 $L_{i,j}^{(t)}$ 和右信息 $R_{i+1,j}^{(t)}$ 更新计算时添加的额外偏移量。

3 改进的左信息更新 RNN-OMS-BP 译码算法

文献[8]中算法通过权重量化机制实现不同迭代次数之前的权值共享,解决了文献[6]中算法乘法运算和缩放因子存储的问题。但该算法仍存在大量的加法和乘法运算,且 BER 性能相较于 DNN-BP 算法有所降低。

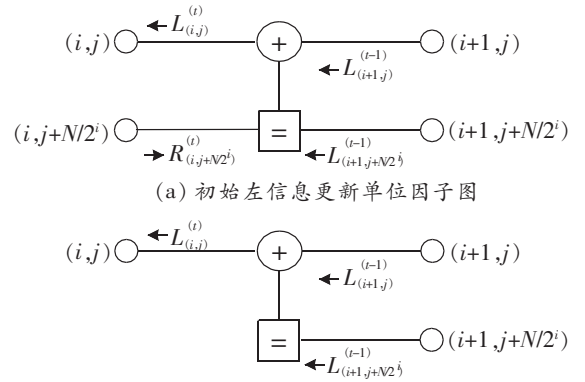
因此,本文提出了一种第 2 节描述的 OMS 替代乘法缩放因子的译码方法解决了乘法运算问题,再根据改进左信息更新的方法和 RNN 架构提出改进的 RNN-OMS-BP-L 译码算法,与文献[8]相比,

算法运算复杂度和迭代次数均有所降低。

3.1 改进的左信息更新方法

本文所提 RNN-OMS-BP-L 译码算法左信息更新的初始单位因子图如图 3(a)所示。

式(9)的 4 个式子是通过软信息处理的思路来进行 LLR 信息的更新计算的,当计算 $L_{i,j}^{(t)}$ 时,在第 1 次迭代中,右更新(R)矩阵只有第 1 列包含信源比特信息,其余列均为 0,因此只关注第 1 列,此时 $L_{i,j}^{(t)}$ 和 $R_{i,j+N/2^i}^{(t)}$ 是独立的。当迭代次数 $T > 1$ 时,其之前的独立性已被消除了,所以在第 2 次及之后的迭代中无需使用该公式进行计算,可将 $R_{i,j+N/2^i}^{(t)}$ 移除,改进的左信息更新方法如图 3(b)所示。



(b) 改进的左信息更新单位因子图

图 3 单位因子图

$L_{i,j}^{(t)}$ 的计算公式为

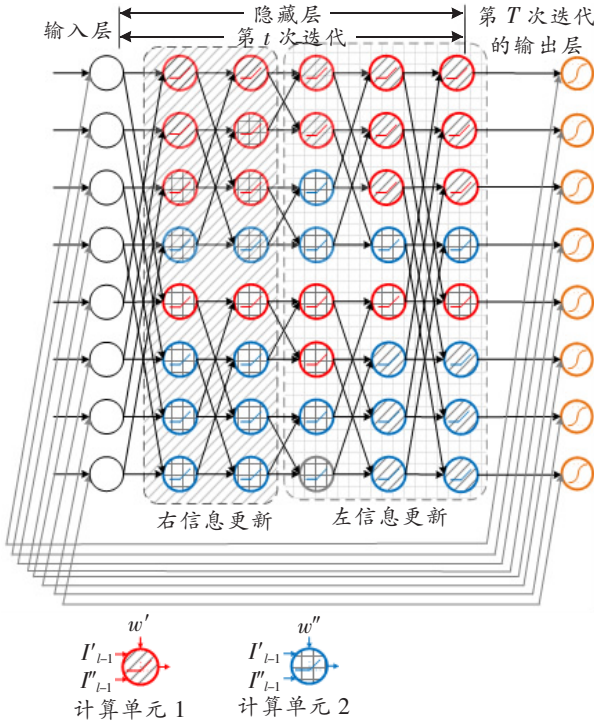
$$L_{i,j}^{(t)} = \begin{cases} g(L_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)} + R_{i,j+N/2^i}^{(t)}, \beta_{L_{i,j}^{(t)}}), & T=1 \\ g(L_{i,j}^{(t)}, L_{i+1,j+N/2^i}^{(t-1)}, \beta_{L_{i,j}^{(t)}}), & T>1 \end{cases}. \quad (10)$$

当迭代次数 $T=1$ 时,本文采用原始更新公式进行左信息更新,当 $T>1$ 时,本文采用式(10)对左信息 $L_{i,j}^{(t)}$ 进行计算。

3.2 改进左信息更新的 RNN 译码器

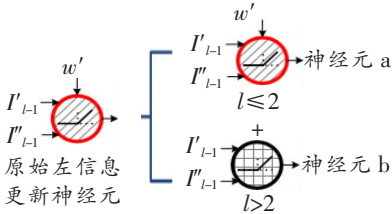
本文基于 3.1 节改进左信息更新方法进而提出了改进的 RNN 架构代替了文献[8]所采用的原始 RNN 架构。图 4(a)所示为 (8,4) 极化码在神经网络中第 t 次迭代的完整信息迭代更新译码过程,其可看作图 2(a)极化码 BP 译码展开形式的因子图,共含 3 层:输入层、隐藏层和输出层。8 个神经计算单元组成输入层;隐藏层对应从左到右的右信息迭代更新过程和反方向的左信息迭代更新过程,分别占据 5 层隐藏层中的第 2 和 3 层神经网络;最后的输出层使用 sigmoid 激活函数来处理当前层的上层更新信息,从而使神经网络译码器迭代更新得到的码字估值归一化。对于 (N,K) 极化码,极化码 BP

因子图展开可得到每层都包含 N 个神经计算单元的 RNN 译码器,其中右迭代信息和左迭代信息更新过程对应第 $n-1$ 层和 n 层隐藏层。



注: I'_{l-1} 和 I''_{l-1} 分别为在第 $l-1$ 层中的两个同类输入信息集; l 为架构层数; w' 和 w'' 分别为两种运算中不同类的输入信息集。

(a) 改进的 (8,4) 极化码 RNN 架构



(b) 改进的左信息更新神经元

图 4 改进的 RNN 译码器

如图 4(a) 所示, RNN 架构隐藏层中包含两种神经计算单元, 其中计算单元 1 用于计算式(9)中的 $L_{i,j}^{(l)}$ 和 $R_{i+1,j}^{(l)}$, 计算单元 2 用于计算 $L_{i,j+N/2^i}^{(l)}$ 和 $R_{i+1,j+N/2^i}^{(l)}$, 分别记为 $f_I(I_{l-1}; \beta_I)^{(l)}$ 和 $f_{II}(I_{l-1}; \beta_{II})^{(l)}$ 。那么式(9)可简化为

$$\begin{cases} f_I(I_{l-1}; \beta_I)^{(l)} = g'(I'_{l-1}, I''_{l-1} + w', \beta_I^I) \\ f_{II}(I_{l-1}; \beta_{II})^{(l)} = g'(I'_{l-1}, w'', \beta_{II}^{II}) + x''_{l-1} \end{cases}, \quad (11)$$

式中: β_I^I 和 β_{II}^{II} 分别为两种运算中的偏移量集。

图 4(b) 所示为本文所提改进的 RNN 相较于文献[9]中原始 RNN 架构的主要改进。本文把原始的左信息更新神经元分成两部分, 当 RNN 架构层数 $l \leq 2$ 时, 使用神经元 a 进行信息更新计算; 当 RNN 架构层数 $l > 2$ 时, 使用神经元 b 进行信息更

新计算, 此时式(11)可更新为

$$\begin{cases} f_I(I_{l-1}; \beta_I)^{(l)} = g'(I'_{l-1}, I''_{l-1} + w', \beta_I^I), l \leq 2 \\ f_I(I_{l-1}; \beta_I)^{(l)} = g'(I'_{l-1}, I''_{l-1}, \beta_I^I), l > 2 \\ f_{II}(I_{l-1}; \beta_{II})^{(l)} = g'(I'_{l-1}, w'', \beta_{II}^{II}) + x''_{l-1} \end{cases}. \quad (12)$$

3.3 RNN-OMS-BP-L 译码算法流程

神经网络译码器完成预置的 T 次迭代更新译码之后, 使用图 4(a) 输出层的 sigmoid 神经单元对 LLR 进行比较判决, 最后再使用交叉熵损失函数衡量改进的 RNN 译码器实际输出值和期望值的差距, 并通过该差距判断译码器的优劣。交叉熵损失函数 $L(u, o)$ 为

$$L(u, o) = -\frac{1}{N} \sum_{j=1}^N u_j \log_2(o_j) + (1 - u_j) \log_2(1 - o_j), \quad (13)$$

式中: N 为码字长度; u_j 和 o_j 分别为改进左信息更新神经网络译码器的理想输出值和实际输出值。基于上述小节提出的改进的左信息更新方法和 RNN 译码器得出的极化码 RNN-OMS-BP-L 算法的具体流程如下:

(1) 设置仿真参数(包括码长 N 、码率 R_c 、信噪比(Signal Noise Ratio, SNR) 范围、训练样本数量 $batch_size$ 和学习率 η 等), 由式(12)生成在随机生成高斯白噪声加扰环境下已知的输入输出映射训练集和验证集。

(2) 由式(4)和式(5)初始化 LLR 信息, 所有非冻结比特的先验 LLR 信息设置为 0, 使用改进 RNN 架构通过式(12)进行 BP 译码, 神经计算单元首先从右至左迭代更新每一列左信息, 直到第 1 列, 再从左至右迭代更新到最后一列, 达到预置的迭代更新次数 T 。

(3) 设置损失函数和优化器, 进行训练和验证, 选择损失最小的最优参数集。

(4) 使用最优参数集的译码器进行测试, 输出生成的实际码字序列, 并与步骤(1)验证集中期望码字序列对比得到损失函数值, 最后计算出 BER。

4 仿真结果与分析

在本节中, 本文所提改进的 RNN 译码器是在 TensorFlow 测试平台上通过学习率为 0.001 的 Adagrad 优化算法训练的。训练数据是在高斯白噪声信道传输后的全零码字, SNR 范围为 1~5 dB。测试数据是随机二进制信息, 该数据是通过极化编码、二进制相移键控(Binary Phase Shift Keying,

BPSK)调制后在 高 斯 白 噪 声 信 道 上 传 输 得 到 的。具 体 参 数 如 表 1 所 示。

表 1 仿真参数设置

参数选项	值
测试平台	TensorFlow
调制方式	BPSK
SNR 范围/dB	1~5
训练集大小	7.5×10^4
测试集大小	3×10^3
优化算法	AdaGrad
损失函数	交叉熵
学习率	0.001
Epochs	25

为了评估本文所提算法的性能,对该算法在不同迭代次数和不同码长的情况下进行了仿真模拟。首先模拟在码长相同和码率都为 1/2 时,不同迭代次数下 BP、RNN-OMS-BP 和 RNN-OMS-BP-L 译码算法的 BER 性能,如图 5 所示。

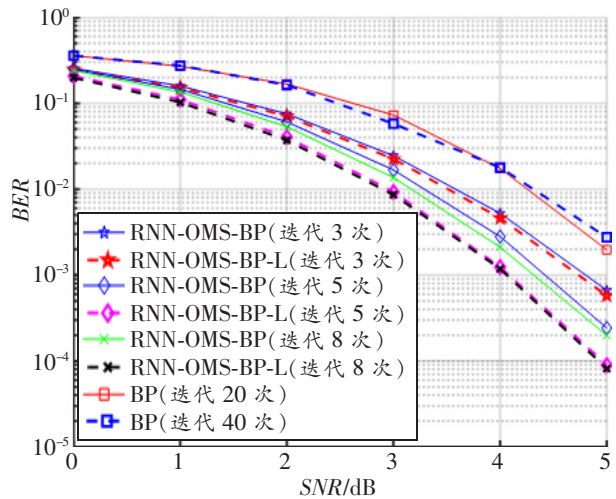


图 5 不同迭代次数下 3 种译码算法 BER 性能对比(以 (64,32) 极化码为例)

由图可知,传统 BP 译码算法迭代 40 次可收敛,而 RNN-OMS-BP-L 和 RNN-OMS-BP 译码算法只需迭代 5~8 次即可收敛,且 BER 性能远优于传统 BP 译码算法。相较于原始 RNN-OMS-BP 译码算法,在相同 BER 性能条件下,本文所提 RNN-OMS-BP-L 译码算法减少了 3 次迭代次数。

第 2 个实验是对极化码 (64,32)、(128,64) 和 (256,128) 3 种不同码长的 DNN-BP、RNN-OMS-BP 和 RNN-OMS-BP-L 译码算法 BER 性能进行仿真对比,迭代次数均为 5 次。实验结果如图 6 所示。由图可知,RNN-OMS-BP-L 译码算法在 3 种不同的码长中,相较于 RNN-OMS-BP 和 DNN-BP 译码

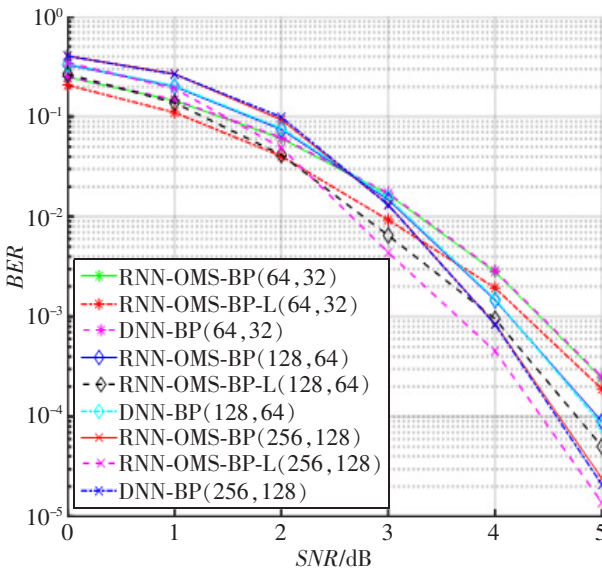


图 6 不同码长下的 3 种算法 BER 性能对比

算法,BER 几乎不变,甚至随着码长的增加,码长为 256 时在高 SNR 区域中极化码 BER 有接近 0.1 dB 的增益。

表 2 所示为以 (64,32) 极化码为例的复杂度分析,列举了在传统 BP 译码算法迭代 $T_{BP} = 40$ 次,DNN-BP、RNN-OMS-BP 译码算法及本文所提 RNN-OMS-BP-L 译码算法迭代 $T = 5$ 次时,分别在加法运算及存储空间占用两个方面的运算复杂度。

表 2 复杂度分析(以 (64,32) 极化码为例)

算法	乘法运算	加法运算	存储空间
BP	0	$2T_{BP}N\log_2 N \sim 30\,720(100\%)$	0
DNN-BP	$2TN\log_2 N$ (100%)	$2TN\log_2 N \sim 3\,840(12.5\%)$	$2TN\log_2 N \sim 3\,840(100\%)$
RNN-OMS-BP	0	$4TN\log_2 N \sim 7\,680(25\%)$	$2N(\log_2 N + 1) \sim 896(23.3\%)$
RNN-OMS-BP-L	0	$3TN\log_2 N \sim 5\,760(18.75\%)$	$2N(\log_2 N + 1) - N/8 \sim 888(23.1\%)$

由表可知,相比于传统 BP 译码算法,改进的 RNN-OMS-BP-L 译码算法将加法运算减少了 81% 以上;相比于 DNN-BP 译码算法,在相同迭代次数条件下,RNN-OMS-BP-L 算法仅以 6.25% 的加法运算为代价替换了全部乘法运算,减少了 76.9% 的存储空间;相比于 RNN-OMS-BP 译码算法,RNN-OMS-BP-L 译码算法在相同 BER 性能条件下,不仅减少了 3 次迭代次数,还减少了 25% 的加法运算和 1% 的存储空间。

5 结束语

本文所提 RNN-OMS-BP-L 译码算法主要是对

信息更新方式的改进,提出了改进的左信息方法和 RNN 架构。通过改变左信息神经计算单元来构建新的 RNN 译码器。由仿真分析可知,改进的左信息更新 RNN 译码方法在几乎无性能损失的前提下,有着运算复杂度和迭代次数的优化,对于 BP 译码算法的硬件实现和实际应用更加友好。接下来将以 RNN-OMS-BP-L 译码算法为基础重点结合神经网络在翻转集的应用^[14-15]方面进行下一步研究。

参考文献:

- [1] Arikan E. Channel Polarization: A Method for Constructing Capacity-achieving Codes[C]//2008 IEEE International Symposium on Information Theory. Toronto, ON, Canada: IEEE, 2008:4595172.
- [2] Niu K, Chen K. CRC-aided Decoding of Polar Codes[J]. IEEE Communications Letters, 2012, 16(10): 1668—1671.
- [3] Tal I, Vardy A. List Decoding of Polar Codes[J]. IEEE Transactions on Information Theory, 2015, 61(5):2213—2226.
- [4] Yuan B, Parhi K K. Architecture Optimizations for BP Polar Decoders[C]//IEEE International Conference on Acoustics. Vancouver, BC, Canada: IEEE, 2013:6638137.
- [5] Yuan B, Parhi K K. Early Stopping Criteria for Energy-Efficient Low-Latency Belief-Propagation Polar Code Decoders[J]. IEEE Transactions on Signal Processing, 2014, 62(24):6496—6506.
- [6] Xu W, Wu Z, Ueng Y L, et al. Improved Polar Decoder based on Deep Learning[C]//2017 IEEE International Workshop on Signal Processing Systems (SiPS). Lorient, France: IEEE, 2017:8109997.
- [7] Nachmani E, Marciano E, Lugosch L, et al. Deep Learning Methods for Improved Decoding of Linear Codes[J]. IEEE Journal of Selected Topics in Signal Processing, 2018,12(1):119—131.
- [8] Teng C F, Wu D C H, Ho K S A, et al. Low Complexity Recurrent Neural Network-based Polar Decoder with Weight Quantization Mechanism[C]//2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Brighton, UK: IEEE, 2019:8683778.
- [9] Doan N, Hashemi S A, Mambou E N, et al. Neural Belief Propagation Decoding of CRC-Polar Concatenated Codes[C]//2019 IEEE International Conference on Communications (ICC). Shanghai, China: IEEE, 2019:8761399.
- [10] 朱旋. SCMA 场景下的极化码编解码方案改进研究[D]. 哈尔滨:哈尔滨工业大学,2019.
- [11] Arikan E. Channel Polarization: A Method for Constructing Capacity-Achieving Codes for Symmetric Binary-Input Memoryless Channels[J]. IEEE Transactions on Information Theory, 2009, 55(7): 3051—3073.
- [12] Wodiany I, Pop A. Low-Precision Neural Network Decoding of Polar Codes[C]//2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). Cannes, France: IEEE, 2019:8815542.
- [13] Lugosch L, Gross W J. Neural Offset Min-Sum Decoding[C]//2017 IEEE International Symposium on Information Theory (ISIT). Aachen, Germany: IEEE, 2017:8006751.
- [14] Sun Y, Shen Y, Song W, et al. LSTM Network-Assisted Belief Propagation Flip Polar Decoder[C]//2020 54th Asilomar Conference on Signals, Systems, and Computers. Pacific Grove, CA, USA: IEEE, 2020: 9443504.
- [15] Teng C, Ho A K, Wu C D, et al. Convolutional Neural Network-Aided Bit-Flipping for Belief Propagation Decoding of Polar Codes[C]//2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). Toronto, ON, Canada: IEEE, 2021: 9413808.