Sets of complementary LLRs to improve OSD post-processing of BP decoding

Joachim Rosseel, Valérian Mannoni, Valentin Savin, Inbar Fijalkow

Sets of complementary LLRs to improve OSD post-processing of BP decoding

Joachim Rosseel*†, Valérian Mannoni*, Valentin Savin*, Inbar Fijalkow†
*CEA-Leti, Université Grenoble Alpes, F-38000 Grenoble, France
{Joachim.Rosseel, Valerian.Mannoni, Valentin.Savin}@cea.fr
†ETIS, CY Cergy Paris Univ., ENSEA, CNRS F-95000, France
{Joachim.Rosseel, Inbar.Fijalkow}@ensea.fr

***Abstract*—This article deals with Ordered Statistics Decoding (OSD) applied to the soft outputs of the Belief Propagation (BP) algorithm. We first model the weighted sum of the a posteriori LLRs across BP decoding iterations into a neuron. The neuron is then trained with the focal loss to compute for each BP decoding failure a set of accumulated Log Likelihood Ratios (LLRs) suited for OSD post-processing. Then, we propose a recursive selection procedure of LLRs sets, for multiple OSD post-processing. This selection is carried out from the sets of a posteriori LLRs calculated at each BP iteration, and from the accumulated LLRs optimized for the OSD, based on their joint probabilities of failure with OSD post-processing. An OSD is then applied to each set of LLRs belonging to the selection. In addition, we propose to reduce the OSD post-processing decoding complexity without significantly degrading its performance. Our results show that this new decoding method provides an effective way to bridge the gap to maximum likelihood decoding for short and long Low Density Parity Check (LDPC) codes.**

***Index Terms*—LDPC, belief propagation, ordered statistics decoding post-processing**

I. Introduction

The emergence of communication systems for the Internet of Things (IoT), with the requirement of short-packet transmissions, has revitalized interest in research and practice of efficient error correcting codes for messages ranging from a few tens up to a few hundred bits. While important progress has been made over the last years in understanding the limits of coding at short block lengths [1], the design of efficient short codes and decoding algorithms still raises many challenges [2].

Low-Density Parity-Check (LDPC) codes [3] are a class of error correcting codes defined by sparse bipartite graphs [4]. They are well-known for their excellent error correction performance at suitably large blocklengths, achieving near Shannon channel capacity performance under iterative belief propagation (BP) decoding [5]. For codes defined by cycle-free bipartite graphs, BP decoding outputs maximum a posteriori estimates of the coded bits [6]. However, finite length codes are actually defined by graphs with cycles, in which case BP decoding is known to be sub-optimal.

In order to improve the error correction capability of short LDPC codes, the BP can be associated with an Ordered Statistics Decoding (OSD) [7]. A first combination strategy, introduced in [8] and used in [9], effectively improves the performance by applying a low order OSD at the end of each BP iteration. However, the resulting complexity is increased significantly due to the use of the OSD at each iteration. To limit the added complexity, the OSD can be applied as a post-processing step, exploiting the BP soft output only when the latter fails to converge to a codeword. As such, the authors of [10] propose an OSD post-processing step for shortened LDPC codes, by considering that some of the information bit values are fixed and known by the receiver. In [11], the accumulation of the Log Likelihood Ratios (LLRs) during BP decoding iterations is used as a new reliability measure of variable-nodes for the OSD. Employing directly the observed LLRs for the reliability measure has been investigated as well in [12]. The OSD post-processing is also carried out in [13] after a decoding diversity composed of several BP modeled with recurrent neural networks (BP-RNNs), each being trained to decode a different kind of error events.

In this paper, we aim firstly at improving the OSD post-processing on the BP outputs for short LDPC codes decoding. Our approach first consists in modeling a weighted sum based on the approach of [11] into a neuron. This neuron is optimized with the focal loss [14] in order to compute for each noisy codeword intended for post-processing a set of accumulated LLRs suited for OSD. Then, we propose to recursively construct a selection of LLRs sets from the sets of a posteriori LLRs computed at each BP decoding iteration, and from the set of the accumulated and optimized LLRs for OSD. To carry out this selection, the complementarity of the considered LLRs sets is measured with their joint probability of failure with OSD post-processing. We thus introduce a new decoding method, where an OSD post-processing is applied to each set of LLRs belonging to the selection. We also show that the proposed decoding strategy above is scalable for longer LDPC codes. Furthermore, reducing the complexity induced by OSD becomes necessary since the number of calculated codewords during OSD tends to be computationally too costly, in particular for medium or large LDPC codes. For this reason, we also propose to reduce the number of candidate codewords computed during the OSD post-processing step.

The paper is organized as follows. Section II introduces the notations, recalls the OSD algorithm and describes the approach of [11]. The training of the neuron modeling the

This work was supported by the French Agence Nationale de la Recherche (ANR), under grant number ANR-21-CE25-0006 (AI4CODE project).

accumulated LLR with the focal loss, the procedure to select complementary sets of LLRs, and the complexity reduction technique are then explained in Section III. Finally, Section IV presents the numerical results, and Section V concludes the paper.

II. OSD post-processing on the a posteriori LLRs of BP decoding

We consider an LDPC code defined by a Tanner (bipartite) graph [4] with N variable-nodes, M check-nodes, denoted respectively by $n \in \{1, \dots, N\}$ and $m \in \{1, \dots, M\}$, and with $K = N - M$ information bits. BP decoding then consists of an iterative exchange of messages along the Tanner graph edges, where each message provides an LLR estimate of the incident variable-node [5].

We also consider the OSD [7], a decoding algorithm capable of approaching the Maximum Likelihood (ML) decoding performance for medium block length linear codes, with polynomial complexity. It can be used as a stand-alone decoder, exploiting the soft channel output, or as post-processing step exploiting the output of a soft decision decoder such as the BP. The first step of the OSD consists to sort the variable-nodes according to a reliability measure, that is the absolute value of the corresponding soft decision. In the case of OSD post-processing for BP, the soft output, and thus the reliability depends of the a posteriori LLR given during BP decoding. The parity check matrix of the code is then brought into a systematic form, denoted by H_{syst}, so that to ensure the K first columns corresponds to the most possible reliable variable-nodes. In OSD-0, the hard decision is taken on the most reliable variable-nodes, and the least reliable ones are determined by solving a linear system given by H_{syst}. Therefore, the decoding is successful if and only if the most reliable variable-nodes are error-free. To deal with cases where these bits contain errors, OSD-p considers all the possible choices of at most p errors among them. For each choice, the initial hard-decision of the corresponding variable-nodes is flipped, and the least reliable variable-nodes are determined again by solving the linear system given by H_{syst}. A list of $\sum_{i=0}^{p} \binom{K}{i}$ codewords, noted by $\mathcal{S}$, is thus computed by this procedure. The most likely codeword in $\mathcal{S}$ is then selected thanks to the ML criterion:

$$\hat{\mathbf{c}} = \arg\max_{\hat{\mathbf{c}}_\mathbf{s} \in \mathcal{S}} P(\hat{\mathbf{c}}_\mathbf{s} | \mathbf{y}) \tag{1}$$

where $\hat{\mathbf{c}}_\mathbf{s}$ represents a codeword determined by OSD-p and $\mathbf{y} = \{y_1, \dots, y_N\}$ the observed (channel) noisy codeword. OSD-p may closely approach the ML decoding performance, assuming the p value (referred to as OSD order) is suitably large.

In order to improve the performance of OSD post-processing with BP soft outputs, the authors of [11] propose to measure the reliability of the variable-nodes for OSD-p with the accumulation of the a posteriori LLRs during BP decoding iterations. More precisely, if the BP fails to obtain a codeword after a maximum number of I iterations, the reliability of a variable-node n is computed as follows:

$$\hat{L}_n^{(\text{S})} = \sum_{i=0}^{I} \hat{L}_n^{(i)}, n \in [1, N] \tag{2}$$

where $\hat{L}_n^{(0)}$ corresponds to the observed LLR of bit n, and $\hat{L}_n^{(i)}$ to its a posteriori LLR calculated at iteration i of BP, for $i \in [1, I]$.

Intuitively, the reliability measure described in equation (2) allows to obtain a stronger reliability for variable-nodes whose $\hat{L}_n^{(i)}$ keep the same sign across BP iterations than those whose $\hat{L}_n^{(i)}$ sign change. Hence, the set $\hat{\mathbf{L}}^{(\text{S})} := \{\hat{L}_1^{(\text{S})}, \dots, \hat{L}_N^{(\text{S})}\}$ ensures that the most reliable variable-nodes and thus fixed by the OSD corresponds to bits for whose BP decoding is "certain" of their values.

III. Multiple OSD post-processing

A. Accumulated and optimized LLR for OSD post-processing

In this section, we are interested in determining an LLR accumulated during BP iterations as suited as possible for OSD post-processing. To do so, we introduce the weights $w^{(i)} \geq 0$, $i \in [0, I]$, in equation (2). We thus propose to compute the reliability of a variable-node n by the following accumulated LLR of I iterations:

$$\hat{L}_n^{(\text{NS})} = \sum_{i=0}^{I} w^{(i)} \hat{L}_n^{(i)}, n \in [1, N] \tag{3}$$

This equation is then modeled by a simple neuron. To create the neuron training set, we first generate a set $\mathcal{T}_{\text{BP}}$ of noisy codewords by considering a binary-input AWGN channel, with a BPSK alphabet (± 1) inputs, and a fixed noise variance σ^2. $\mathcal{T}_{\text{BP}}$ is then decoded by the BP. Since the AWGN channel and BP are symmetrical, only the all-zero codeword is transmitted. We subsequently denote by $\mathcal{T}_{\text{BP-OSD}}$ the subset of noisy codewords of $\mathcal{T}_{\text{BP}}$ for which the BP does not find a codeword after I iterations. The sets $\hat{\mathbf{L}}_\mathbf{n} := \{\hat{L}_n^{(0)}, \dots, \hat{L}_n^{(I)}\}$, (with $n \in [1, N]$) of each noisy codeword belonging to $\mathcal{T}_{\text{BP-OSD}}$ then constitute the training set of the neuron. This training set allows the neuron to be trained only in cases where the BP does not converge to a codeword.

To optimize the neural weights, we propose to utilize the focal loss, first introduced in [14], and defined by:

$$\begin{aligned} \text{FL}\big(b_n, \hat{L}_n^{(\text{NS})}\big) = &-b_n \sigma\big(\hat{L}_n^{(\text{NS})}\big)^\gamma \log\Big(1 - \sigma\big(\hat{L}_n^{(\text{NS})}\big)\Big) \\ &- (1 - b_n)\Big(1 - \sigma\big(\hat{L}_n^{(\text{NS})}\big)\Big)^\gamma \log\Big(\sigma\big(\hat{L}_n^{(\text{NS})}\big)\Big) \end{aligned} \tag{4}$$

where b_n is the expected value of bit n, $\sigma(x) = (1 + e^{-x})^{-1}$ is the sigmoid function converting the LLR value into the probability that the decoded bit is equal to zero, and $\gamma \geq 0$ is an an adjustable hyper-parameter. By assuming that the all-zero word is transmitted, that is $b_n = 0$ for $n \in [1, N]$, (4) simplifies to:

$$\text{FL}\big(\hat{L}_n^{(\text{NS})}\big) = -\Big(1 - \sigma\big(\hat{L}_n^{(\text{NS})}\big)\Big)^\gamma \log\Big(\sigma\big(\hat{L}_n^{(\text{NS})}\big)\Big) \tag{5}$$

The focal loss enables to focus the training on the hardest elements to classify in the training set, by affecting them higher penalties. In our case, the most difficult vectors to classify are the $\hat{\mathbf{L}}_{\mathbf{n}}$ possessing $\hat{L}_n^{(i)}$ with larges amplitudes and incorrect signs, since it will result in a $\hat{L}_n^{(\text{NS})}$ with the same characteristics. Minimizing the focal loss therefore optimizes the weights to reduce the impact of such $\hat{L}_n^{(i)}$ on the computation of $\hat{L}_n^{(\text{NS})}$. Moreover, resulting $\hat{L}_n^{(\text{NS})}$ with large amplitudes and incorrect signs induce erroneous bits, which will be potentially selected among the K most reliable ones during OSD. Consequently, minimizing the focal loss reduces the probability of having erroneous bits among the K the most reliable bits. $\hat{\mathbf{L}}^{(\text{NS})}$ is thus effectively optimized for OSD post-processing. To the best of our knowledge, it is the first time the focal loss [14] is used in decoding. We noticed that training an LLR $\hat{\mathbf{L}}^{(\text{NS})}$ with a focal cost $\gamma > 0$ outperforms the training with a binary cross entropy (that is $\gamma = 0$) when OSD post-processing is applied.

B. Selecting sets of LLRs

In order to get closer to the ML decoding performance, we propose here a decoding method where an OSD post-processing is applied after each LLRs set of an ordered list $\mathcal{L}_Z$ of Z LLRs sets, with $Z \in [1, I+2]$. To construct this list, the complementarity of $\hat{\mathbf{L}}^{(\mathbf{i})} := \{\hat{L}_1^{(i)}, \ldots, \hat{L}_N^{(i)}\}$, $i \in [0, I]$, and of $\hat{\mathbf{L}}^{(\text{NS})}$ with OSD is evaluated with the number of errors remaining after OSD post-processing.

We first generate a test set $\mathcal{T}_{\text{BP-OSD}}$, according to the procedure described in the previous section. The decoding performance of the OSD post-processing is then assessed on $\mathcal{T}_{\text{BP-OSD}}$ with each $\hat{\mathbf{L}}^{(\mathbf{i})}$ and with $\hat{\mathbf{L}}^{(\text{NS})}$. We denote by $\mathcal{F}^{(i)} \subset \mathcal{T}_{\text{BP-OSD}}$ (resp. $\mathcal{F}^{(\text{NS})} \subset \mathcal{T}_{\text{BP-OSD}}$) the subset of noisy words on which $\hat{\mathbf{L}}^{(\mathbf{i})}$ (resp. $\hat{\mathbf{L}}^{(\text{NS})}$) leads to a failure during OSD-p decoding. Then, we recursively construct an ordered list of LLRs sets, noted $\mathcal{L}$. This list is initialized with $\hat{L}^{(\text{NS})}$, $\mathcal{L} = \left\{\hat{\mathbf{L}}^{(\text{NS})}\right\}$, since $\hat{\mathbf{L}}^{(\text{NS})}$ is optimized for OSD post-processing. To add a new LLRs set $\hat{\mathbf{L}}^{\text{new}}$ to $\mathcal{L}$, we propose to apply the following rule:

$$\hat{\mathbf{L}}^{\text{new}} = \underset{\hat{\mathbf{L}}^{(\mathbf{i})} \in \{\hat{\mathbf{L}}^{(\mathbf{0})}, \ldots, \hat{\mathbf{L}}^{(\mathbf{I})}\} \setminus \mathcal{L}}{\arg\min} \left| \mathcal{F}_{\mathcal{L}} \cap \mathcal{F}^{(i)} \right|, \tag{6}$$

where $\mathcal{F}_{\mathcal{L}} := \mathcal{F}^{(\text{NS})}$ if $\mathcal{L} = \left\{\hat{\mathbf{L}}^{(\text{NS})}\right\}$, $\mathcal{F}_{\mathcal{L}} := \mathcal{F}^{(\text{NS})} \cap \left(\cap_{\hat{\mathbf{L}}^{(\mathbf{i})} \in \mathcal{L}} \mathcal{F}^{(i)} \right)$ otherwise. The above rule is applied $I+1$ times, until $\mathcal{L}$ contains all the LLRs sets. If the minimum argument of (6) is not unique, an arbitrary choice is made among these values.

For $Z \leq I+2$, $\mathcal{L}_Z$ is defined as the sub-list of the Z first LLRs sets of $\mathcal{L}$. $\mathcal{L}_Z$ is thus an ordered list of LLRs sets, representing Z complementary levels of reliability with respect to OSD post-processing. As the OSD post-processing is applied to each element of $\mathcal{L}_Z$, an ML rule such as defined in (1) is used to choose the final codeword among the Z candidate codewords. In the following, we note this decoding method by BP-$\mathcal{L}_Z$-OSD-p.

Finally, we point out that neither the neural modeling of the weighted sum, nor the construction of $\mathcal{L}_Z$, depend of the dimensions N and K of the considered LDPC code. As a result, the methodology presented in this paper is reproducible for any LDPC code.

C. Complexity reduction

A main practical limitation for the OSD implementation is its decoding complexity [15], since a list of $\sum_{i=0}^{p} \binom{K}{i}$ candidate codewords has to be calculated at each OSD utilisation. Hence, the OSD post-processing complexity tends to be especially costly for medium or long LDPC codes with $K > 100$ and for OSD-p with $p \geq 2$. To address this issue, the number of candidate codewords is thus usually limited by flipping only some of the most reliable variable-nodes. As such, a procedure to compute a list of positions to be flipped during OSD-p was introduced in [15]. This list is notably determined thanks to the computation of joint error probabilities for the most reliable bits, and allows to process only the most probable candidate codewords. In [16], the most reliable bits are partitioned into segments according to reliability thresholds depending of the received noisy codeword. Only some of the segments are then selected for flipping. Both previously described methods can require a high computational cost, due in particular to the threshold computations. The authors of [17] propose for each noisy codeword to flip only the most reliable variable-nodes for which soft values do not respect amplitude thresholds.

Here, we propose a strategy to reduce the computational complexity, which does not depend of the noisy codeword. More precisely, we first put a limitation to the decoding complexity by restricting our study to OSD-2. We then reduce the complexity of OSD-2, and by extension of BP-$\mathcal{L}_Z$-OSD-2, by limiting the choices to a maximum of two errors according to their level of reliability. We thus introduce two positions thresholds, T_1 and T_2, with $T_1 < T_2$. For simplicity, the most reliable bits are also numbered from 0 to K in ascending order of reliability. For a linear code, we thus consider only up to two flips between $[0, T_2]$, with at least one flip in $[0, T_1]$ in the case of two flips. Note that the bits with sorting positions in the segment $[T_2+1, K]$ correspond to bits with the highest reliability, and thus with the lowest probability of error. In addition, a couple of errors in $[T_1+1, T_2]^2$ is less probable than a couple of errors in $[0, T_1]^2$ or in $[0, T_1] \times [0, T_2]$. Therefore, the proposed method limits the number of candidate codewords for OSD-2 by processing only error events with the highest probability of occurrence. The number of candidate codewords N_C is obtained with the following formula:

$$N_C = 1 + \binom{T_2}{1} + \binom{T_1}{2} + (T_2 - T_1)T_1 \tag{7}$$

The optimal choice of T_1 and T_2 is then determined by a complexity/performance evaluation thanks to (7) . Indeed, for a fixed number of candidate codewords N_c, all the couples satisfying (7) are determined. We then assessed the FER of BP-$\mathcal{L}_Z$-OSD-2 at a fixed SNR with each found couple. The couple for which the reduced complexity BP-$\mathcal{L}_Z$-OSD-2 obtains the best FER performance is thus selected.

Finally, note that for BP-$\mathcal{L}_Z$-OSD-p, the number of tested codewords when the BP fails is equal to $Z \times \sum_{i=0}^{p} \binom{K}{i}$ since Z OSD-p are performed. Therefore, the value of Z may also be limited for complexity reasons as it directly impact the number of tested codewords and the number of system resolutions.

IV. Numerical results

A. Simulation settings

Three LDPC codes have been considered in our simulations. Their parameters are provided in Table I, where $R_c := K/N$ denotes the coding rate, d_v the variable-nodes degree, and d_c the check-nodes degree.

TABLE I
LDPC codes parameters

	$\mathbf{N}$	$\mathbf{K}$	$\mathbf{R_c}$	$\mathbf{d_v}$	$\mathbf{d_c}$
CCSDS code [18]	128	64	0.5	3-5	8
Tanner code [19]	155	64	0.41	3	5
MacKay code [20]	1008	504	0.5	3	6

The number of BP decoding iterations was set to 25 for the CCSDS and Tanner codes, and to 10 for the MacKay code. The OSD-p post-processing was evaluated for $p = 0, 1, 2$ in order to limit the number of tested combinations. Concerning the neuron introduced in Section III-A, a set $\mathcal{T}_{\text{BP-OSD}}$ of 10000 noisy codewords was generated to create the training set. The weights, all initialized to one, were then optimized over 50 epochs, so that the focal loss converge towards a limit. Furthermore, we assessed the Frame Error Rate (FER) of the OSD-p with $\hat{\mathbf{L}}^{(\text{NS})}$ according to the hyper-parameter γ, and we determined empirically $\gamma = 10$ as being a good choice to penalize the misclassified LLRs. Finally, the neuron was trained for each SNR value ranging from $2.5\,\text{dB}$ to $4.5\,\text{dB}$ (resp. from $1.5\,\text{dB}$ to $3.5\,\text{dB}$), with a step of $0.5\,\text{dB}$ for the CCSDS code (resp. Tanner code/MacKay code). In addition, for each SNR value, a test set $\mathcal{T}_{\text{BP-OSD}}$ of 10000 noisy codewords was generated, and a list $\mathcal{L}$ was thereby constructed according to the procedure described in section III-B. We compare then the different decoding strategies proposed in this paper for the three codes, in terms of FER. Reported SNR gains are evaluated at a FER of 10^{-4}.

B. CCSDS Code

On Fig. 1, the performance of CCSDS decoding for an OSD-p post processing with $\hat{\mathbf{L}}^{(\text{NS})}$ is compared with the performance of an OSD-p post processing using $\hat{\mathbf{L}}^{(\text{S})}$ as a reliability measure. We observe with $\hat{\mathbf{L}}^{(\text{NS})}$ a slightly better performance than $\hat{\mathbf{L}}^{(\text{S})}$ for $p = 1$ (similar for $p = 0$). For an order $p = 2$, $\hat{\mathbf{L}}^{(\text{NS})}$ allows to obtain a gain of $0.16\,\text{dB}$ with respect to $\hat{\mathbf{L}}^{(\text{S})}$. As a result, $\hat{\mathbf{L}}^{(\text{NS})}$ becomes more and more suited to OSD when the OSD order increases.

In the following, we consider a maximum budget of 3 OSD-p post-processing. To start with, the set $\mathcal{L}_3$ is determined from $\mathcal{L}$, as explained in section III-B. The performance of OSD-p post-processing with $\mathcal{L}_3$ and with $\hat{\mathbf{L}}^{(\text{NS})}$ alone are illustrated in Fig. 2. We notice that applying an OSD-p after each LLRs set of $\mathcal{L}_3$ provides an increasing gain with respect to the order p. Indeed, the gain is respectively of $0.12\,\text{dB}$, $0.22\,\text{dB}$, and

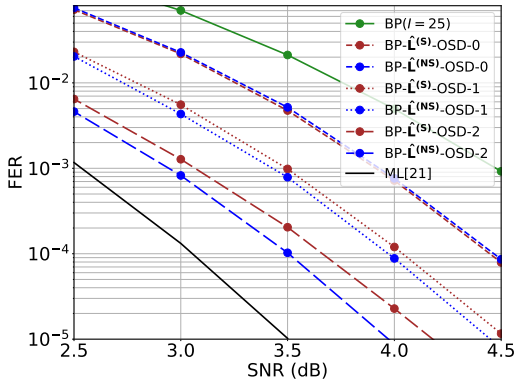

Fig. 1. FER for CCSDS code, $\hat{L}^{(\text{S})}$ vs $\hat{L}^{(\text{NS})}$.

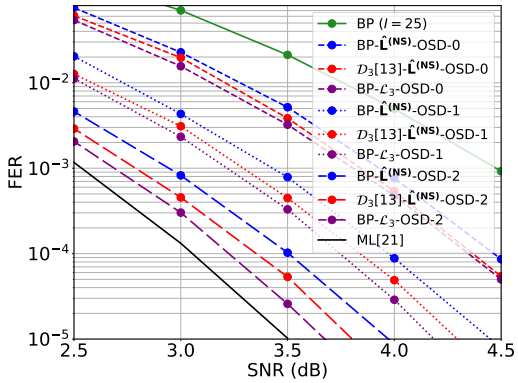

Fig. 2. FER for CCSDS code.

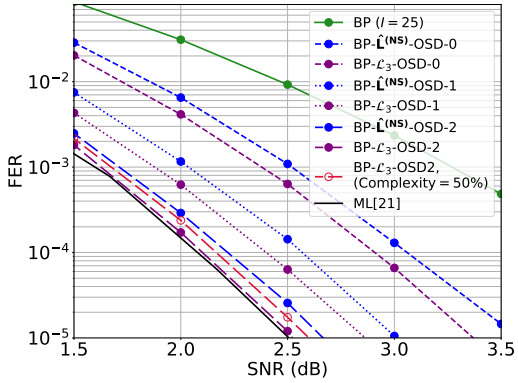

Fig. 3. FER for Tanner code.

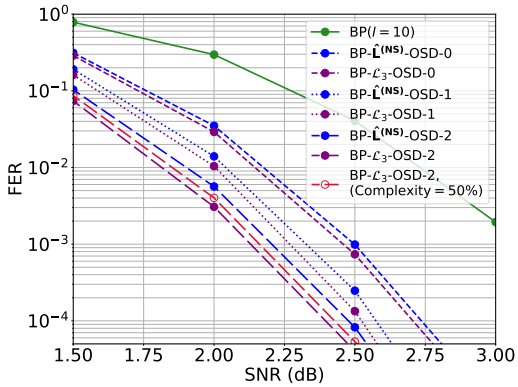

Fig. 4. FER for MacKay code.

$0.27\,\mathrm{dB}$ for $p = 0, 1, 2$. In addition, it can be observed that BP-$\mathcal{L}_3$-OSD-2 achieves a FER performance at only $0.17\,\mathrm{dB}$ from ML decoding [21].

Finally, OSD-p post-processing is applied after a decoding diversity of 3 BP-RNNs, denoted $\mathcal{D}_3$. The construction method of this decoding diversity is detailed in [13]. For each BP-RNN of $\mathcal{D}_3$, a single neuron is optimized with the parameters described in section IV-A. Three $\hat{\mathbf{L}}^{(\mathrm{NS})}$ reliability are thus computed and assessed with an OSD-p post-processing. An ML rule decides of the the final codeword. We note this diversity approach by $\mathcal{D}_3$-$\hat{\mathbf{L}}^{(\mathrm{NS})}$-OSD-$p$, and the corresponding results are shown in Fig. 2. It can be observed that BP-$\mathcal{L}_3$-OSD-0 provides a slight improvement over $\mathcal{D}_3$-$\hat{\mathbf{L}}^{(\mathrm{NS})}$-OSD-0. The gain increases to $0.1\,\mathrm{dB}$ for $p = 1$, and then to $0.12\,\mathrm{dB}$ for $p = 2$. As a result, using the BP decoder alone and constructing a list of complementary LLRs sets is a better strategy in term of FER performance with OSD-p post-processing.

C. Tanner Code

The simulation results obtained with the Tanner code are presented on Fig. 3, with the same methodology. The decoding by BP-$\mathcal{L}_3$-OSD-0 provides a gain of $0.16\,\mathrm{dB}$ with respect to BP-$\hat{\mathbf{L}}^{(\mathrm{NS})}$-OSD-0. This gain remains similar when the OSD order is 1 or 2. Furthermore, we observe that BP-$\mathcal{L}_3$-OSD-2 nearly reaches the ML decoding performance.

For this code, we also consider a decoder BP-$\mathcal{L}_3$-OSD-2 operating with a complexity reduced by 50%, as described in Section III-C. This complexity reduction amounts to $N_C = 3121$ instead of the BP-$\mathcal{L}_3$-OSD-2 $3 \times \sum_{i=0}^{2} \binom{64}{i} = 6243$ codewords tested at each BP decoding failure. An optimal threshold couple (T_1, T_2) for BP-$\mathcal{L}_3$-OSD-2 is thus determined for each SNR value. The corresponding performance is illustrated in Fig. 3. We notice that the complexity reduction of 50% induces a degradation of only $0.08\,\mathrm{dB}$ with respect to BP-$\mathcal{L}_3$-OSD-2 with no reduction.

D. Extension to longer code: MacKay code

Fig. 4 shows the simulations results for the MacKay code. This code possesses higher dimensions N and K than both previously discussed codes, but as explained in Section III-B, the methodology is reproducible. It can be observed that BP-$\mathcal{L}_3$-OSD-p exhibits small improvements with respect to BP-$\hat{\mathbf{L}}^{(\mathrm{NS})}$-OSD-$p$, up to $0.1\,\mathrm{dB}$ for an order $p = 2$.

However, since $K = 504$, $N_c = 381783$ candidate codewords are computed at each BP decoding failure for BP-$\mathcal{L}_3$-OSD-2. Consequently, a complexity reduction becomes mandatory for the MacKay code. As such, the decoder BP-$\mathcal{L}_3$-OSD-2 with a complexity reduction of 50% is assessed. An optimal threshold couple is hence calculated for each SNR value. We observe that the decoder BP-$\mathcal{L}_3$-OSD-2 with a complexity reduction of 50% tends to be almost as efficient than the standard BP-$\mathcal{L}_3$-OSD-2 decoder.

V. Conclusion

In this article, we addressed the problem of enhancing LDPC codes decoding with an OSD post-processing on the BP outputs. To this end, we optimized an accumulated LLR for OSD post-processing using the focal loss. We then built recursively an ordered list of complementary LLRs sets with respect to the OSD algorithm. This list allowed us to propose a new decoding strategy, where an OSD post-processing is applied after each LLRs set of the list. Finally, we showed that this methodology can be reproduced for any codeword length, provided that the complexity of the OSD is limited. This work also opens new perspectives on the utilisation of the focal loss for decoding with neural networks.

References

[1] Y. Polyanskiy, H. V. Poor, and S. Verdú, "Channel coding rate in the finite blocklength regime," *IEEE Trans. on Inf. Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.

[2] M. C. Coşkun, G. Durisi, T. Jerkovits, G. Liva, W. Ryan, B. Stein, and F. Steiner, "Efficient error-correcting codes in the short blocklength regime," *Physical Communication*, vol. 34, pp. 66–79, 2019.

[3] R. G. Gallager, "Low density parity check codes," MIT Press, Cambridge, 1963, research Monograph series.

[4] R. Tanner, "A recursive approach to low complexity codes," *IEEE Trans. on Inf. Theory*, vol. 27, no. 5, pp. 533–547, 1981.

[5] T. Richardson, M. Shokrollahi, and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes," *IEEE Trans. on Inf. Theory*, vol. 47, no. 2, pp. 619–637, 2001.

[6] N. Wiberg, "Codes and decoding on general graphs," Ph.D. dissertation, Department of Electrical Engineering, Linköping University, Sweden, 1996.

[7] M. Fossorier and S. L., "Soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. on Inf. Theory*, vol. 41, no. 5, pp. 1379–1396, 1995.

[8] M. Fossorier, "Iterative reliability-based decoding of low-density parity check codes," *IEEE Journal on Selected Areas in Com.*, vol. 19, no. 5, pp. 908–917, 2001.

[9] W. Zhou and M. Lentmaier, "Improving short-length LDPC codes with a CRC and iterative ordered statistic decoding," in *CISS*, 2019, pp. 1–6.

[10] K. Watanabe, R. Kaguchi, and T. Shinoda, "Shortened LDPC codes accelerate OSD decoding performance," *EURASIP Journal on Wireless Com. and Networking*, vol. 2021, no. 1, pp. 1–18, 2021.

[11] M. Jiang, C. Zhao, E. Xu, and L. Zhang, "Reliability-based iterative decoding of LDPC codes using likelihood accumulation," *IEEE Com. letters*, vol. 11, no. 8, pp. 677–679, 2007.

[12] M. Baldi, N. Maturo, E. Paolini, and F. Chiaraluce, "On the applicability of the most reliable basis algorithm for ldpc decoding in telecommand links," in *Int. Conference on Inf. and Com. Systems (ICICS)*. IEEE, 2015, pp. 1–6.

[13] J. Rosseel, V. Mannoni, I. Fijalkow, and V. Savin, "Decoding short LDPC codes via BP-RNN diversity and reliability-based post-processing," *IEEE Trans. on Com.*, vol. 70, no. 12, pp. 7830–7842, 2022.

[14] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[15] M. P. C. Fossorier and S. Lin, "Computationally efficient soft-decision decoding of linear block codes based on ordered statistics," *IEEE Trans. on Inf. Theory*, vol. 42, no. 3, pp. 738–750, 1996.

[16] C. Yue, M. Shirvanimoghaddam, Y. Li, and B. Vucetic, "Segmentation-discarding ordered-statistic decoding for linear block codes," in *2019 IEEE GLOBECOM*. IEEE, 2019, pp. 1–6.

[17] D. Wu, Y. Li, X. Guo, and Y. Sun, "Ordered statistic decoding for short polar codes," *IEEE Com. Letters*, vol. 20, no. 6, 2016.

[18] "Short block length LDPC codes for TC synchronization and channel codding (CCSDS 231.1-O-1)," Consultative Committee for Space Data Systems (CCSDS), Techical Report, April 2015.

[19] R. Tanner, D. Sridhara, and T. Fuja, "A class of group-structured LDPC codes," in *Proc. ISTA*. Citeseer, 2001, pp. 365–370.

[20] D. J. C. MacKay, "Encyclopedia of sparse graph codes," http://www.inference.org.uk/mackay/codes/data.html, 2008.

[21] M. Helmling, S. Scholl, F. Gensheimer, T. Dietz, K. Kraft, S. Ruzika, and N. Wehn, "Database of channel codes and ML simulation results," www.uni-kl.de/channel-codes, 2019.