Friendly Attacks to Improve Channel Coding Reliability

Anastasiia Kurmukova and Deniz Gunduz

Department of Electrical and Electronic Engineering, Imperial College London, London, UK
a.kurmukova22@imperial.ac.uk, d.gunduz@imperial.ac.uk

Abstract

This paper introduces a novel approach called "friendly attack" aimed at enhancing the performance of error correction channel codes. Inspired by the concept of adversarial attacks, our method leverages the idea of introducing slight perturbations to the neural network input, resulting in a substantial impact on the network's performance. By introducing small perturbations to fixed-point modulated codewords before transmission, we effectively improve the decoder's performance without violating the input power constraint. The perturbation design is accomplished by a modified iterative fast gradient method. This study investigates various decoder architectures suitable for computing gradients to obtain the desired perturbations. Specifically, we consider belief propagation (BP) for LDPC codes; the error correcting code transformer, BP and neural BP (NBP) for polar codes, and neural BCJR for convolutional codes. We demonstrate that the proposed friendly attack method can improve the reliability across different channels, modulations, codes, and decoders. This method allows us to increase the reliability of communication with a legacy receiver by simply modifying the transmitted codeword appropriately.

Introduction

Channel coding plays a crucial role in modern communication systems, ensuring reliable information transmission over noisy channels. Over the past 70 years, remarkable progress has fueled the continuous development of this field, yielding robust and efficient communication protocols that employ a wide range of hand-crafted codes. Contemporary standards such as Long Term Evolution (LTE) and 5G (3GPP 2018) have been significantly influenced by the development of near-optimal channel codes, including linear block codes such as polar codes (Arikan 2008) and low density parity check (LDPC) codes (Gallager 1962), alongside linear codes with memory, such as turbo codes (Berrou, Glavieux, and Thitimajshima 1993) based on convolutional codes (Elias 1955). These codes demonstrate capacity-approaching performance (Shannon 1948) on additive white Gaussian noise (AWGN) channels; however, their optimal decoding is still a challenging problem.

Despite progress, a gap persists between achievability bounds based on random coding (Polyanskiy, Poor, and Verdú 2010) and the performance of hand-crafted codes in the short block length regime. Maximum likelihood decoding of linear codes is known to be NP-hard. Iterative algorithms, such as belief propagation (BP) operating over the code Tanner graph, present an efficient alternative. However, BP decoding demonstrates near-capacity performance primarily for LDPC codes with large block lengths (MacKay and Neal 1997). For polar codes, BP decoding performance lags behind successive cancellation (SC) decoding (Arikan 2008) attributed to a great number of short cycles in their Tanner graphs.

Decoding of channel codes can be considered as a classification problem, where the aim is to infer the transmitted codeword from its noisy version. Therefore, the recent success of deep learning methods for a wide range of classification problems and their ability to capture complex patterns have motivated exploring their applications to channel decoding. Neural BP (NBP), which introduces trainable weights to conventional BP graphs, has gained popularity due to its simplicity and efficiency (Nachmani, Be'ery, and Burshtein 2016; Nachmani et al. 2018; Lugosch and Gross 2017). Neural BP decoding of polar code has enhanced the code performance (Xu et al. 2017; Doan et al. 2018a). Recently, a significant improvement is achieved in NBP decoding with two-stage decimation (Buchberger et al. 2020).

There are other model-based neural decoding approaches that leverage existing non-neural decoders' structure to enhance performance with trainable architectures. Notably, neural successive cancellation for polar codes (Doan, Ali Hashemi, and Gross 2018), neural BCJR (NBCJR) for convolutional codes (Kim et al. 2018), KO-codes for Reed-Muller codes (Makkuva et al. 2021) have shown promising results. Conversely, model-free deep learning decoding approaches, like (Gruber et al. 2017) and (O'Shea and Hoydis 2017), which do not use any existing decoding algorithm as a baseline, are limited to short block lengths ($k < 16$) due to the curse of dimensionality. Syndrome-based schemes (Bennatan, Choukroun, and Kisilev 2018) and the error correction code transformer (ECCT) (Choukroun and Wolf 2022) have recently achieved performance improvements by embedding the code structure through the syndromes into the learnable decoding architecture.

The improvement in transmission performance extends beyond the enhancement of channel decoders. In real-world transmission, fixed-constellation modulation is usually em-

ployed, involving a uniform mapping of a bit sequence (or a codeword) into a sequence of symbols from a finite set of constellation points. Examples of such modulations include commonly used binary phase shift keying (BPSK) modulation (with constellation $\{1, -1\}$) and higher order modulations like quadrature phase shift keying (QPSK), 2^n quadrature amplitude modulation (QAM). It was shown that the traditional choice of the constellation may not be optimal for various communication systems, therefore the modulation performance can be enhanced by the technique called signal shaping (Qu and Djordjevic 2019). There are two prevalent approaches allowing the selection of more suitable modulations for transmission: geometrical shaping and probabilistic shaping. The first method involves locating optimal transmission constellation points according to specific criteria, while the second method adjusts the probabilities of transmitted symbols to maximize the mutual information between the channel input and the output.

Learning based techniques employing an autoencoder architecture have also been used for geometric shaping (O'Shea and Hoydis 2017; Jones, Yankov, and Zibar 2019; Gümüş et al. 2020) as well as joint geometric and probabilistic shaping (Aref and Chagnon 2022). While they provide considerable gains, they rely on modifying both the encoder and decoder architectures, which may not be viable in some practical systems. In contrast, the goal in this work is to increase the reliability of communication with a fixed oblivious receiver, by only modifying the transmitted signals. This can allow, for example, higher reliability in the downlink of cellular communication networks or digital radio/TV broadcasting systems without upgrading mobile devices.

We treat the modifications to the transmitted symbols as perturbations to the modulated codewords. The goal is to identify the ideal perturbations that are more likely to result in correct decoding given a fixed decoding rule, e.g., fixed decoding boundaries. This can be considered as the converse of adversarial attacks on neural networks (Goodfellow, Shlens, and Szegedy 2015). We propose a new concept of 'friendly attacks' for channel encoding, where the goal is to find a perturbation to the modulated codeword that improves the decoding performance without violating the average power constraint at the transmitter.

The underlying idea of a 'friendly attack' is as follows. The space of the codewords is divided by the decoder into decision regions. Modulated codewords may be suboptimal for these decision regions, particularly when the decoding algorithm is suboptimal. The optimal codeword locations should be as far from the decision boundaries as possible. Hence, a friendly attack seeks a perturbation vector (or, an attack vector) that shifts the modulated codeword towards the center of the corresponding decision region.

We design the attack vector by finding gradients during noisy channel transmission and subsequent decoding, akin to white-box adversarial attacks (Goodfellow, Shlens, and Szegedy 2015; Kurakin, Goodfellow, and Bengio 2016, 2017). The computed attack vector is added to the modulated codeword before transmission. We highlight the differences of friendly attacks compared to conventional adversarial attacks on neural networks: The most obvious difference is that our goal is to improve the quality of decisions made by the receiver, while adversarial attacks try to fool the neural network. Moreover, in adversarial attacks, we determine a specific attack for a specific input that deterministically results in a wrong decision. In the case of channel decoding, codewords are decoded correctly in most cases, while errors happen rarely, when the random channel noise moves the received codeword to the wrong decision region. Hence, our goal is to modify the channel input to increase the correct decoding probability. Finally, in adversarial attacks, perturbations are bounded in order to limit their perception by humans. Here, perturbations are bounded to guarantee that the average power of the transmitted codeword remains the same after its application. We also would like to emphasize that, thanks to the linearity of the code, there is no need to find a distinct specific perturbation for each codeword. It is sufficient to find a perturbation for the all-zero codeword, which can then be applied to any codeword resulting in a similar improvement to the decoding performance.

The friendly attack approach to channel coding proposed here is applicable to any code, and any neural or differentiable decoding algorithm. We provide the results of successful friendly attacks for following scenarios: LDPC code $n = 64, k = 32$ for BP decoder with BPSK and 4-QAM modulations; polar code $n = 64, k = 32$ for BP, NBP and ECCT decoders with BPSK and 4-QAM modulations and also for fading channel with BPSK; long block length polar code with $n = 512, k = 256$ for BP and NBP with BPSK modulation; convolutional code $k = 100, R = \frac{1}{2}$ with NBCJR decoder over AWGN and bursty channels. We show that the proposed friendly attack can improve the decoding performance significantly for a suboptimal decoder in the high SNR regime. Our results show that thoroughly designed attack vector enhancing BP decoding for a chosen code can also improve the performance of a BP decoder with trainable weights for the same code and the same number of iterations. Moreover, for the considered codes the improvement by friendly attack for the BER of BP decoding with several iterations is greater than improvement achieved by adding trainable weights to BP decoding (NBP). The idea of perturbing the modulated codeword with an attack vector was also proposed in (Hameed, György, and Gündüz 2021), but there the goal is to prevent an eavesdropper trying to detect the modulation scheme of the transmitted waveform; hence, it follows the conventional adversarial attack framework.

Background

We use the following notations for vectors: $\mathbf{x}^n = \{x_1, x_2, \ldots, x_n\}$, $\mathbf{x}_i^j = \{x_i, x_{i+1}, \ldots, x_j\}$; a bold notation $\mathbf{x}^n$ for a vector with fixed values and X^n for the corresponding vector of random variables.

Communication Scheme

Communication scheme, as shown in Fig. 1, utilizes a channel code for encoding information at the transmitter and decoding information from a noisy sequence at the receiver. For encoding and decoding we assume traditional transmission setting with a linear (n, k) code $\mathcal{C}$ over a binary field

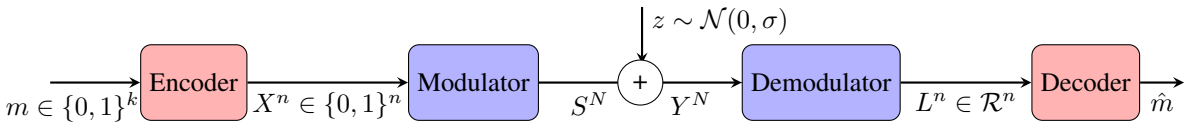

Figure 1: The studied coded communication scenario with AWGN channel.

$\mathbb{F} = \{0, 1\}$. The sender has a k-bit message $m \in \{0, 1\}^k$, which is encoded into a codeword X^n from the code space $\mathcal{C}$. The modulator then maps the sequence of codeword bits into a sequence of symbols $\mathbf{s}^N$ from a chosen discrete constellation, and the modulated codeword is transmitted over the physical channel. For instance, BPSK modulation has the constellation $\{1, -1\}$, and the modulation can be written as:

$$S = 1 - 2X \, .$$

We will also consider modulations with a higher constellation order such as 4-QAM. Here we suppose that the communication scheme must follow the average power constraint for the codeword. Every modulated codeword $\mathbf{s}^N$ must satisfy average power constraint P: $||\mathbf{s}^N||_2^2 \leq NP$.

We consider an additive white Gaussian noise (AWGN) channel given as follows:

$$Y^N = S^N + Z^N,$$

where Z^N is a sequence of independent and identically distributed (i.i.d.) Gaussian random variables $Z_i \sim \mathcal{N}(0, \sigma^2)$. We will also consider a Rayleigh fading channel (Hou, Siegel, and Milstein 2001), where

$$Y^N = G^N S^N + Z^N,$$

where channel gains G^N are i.i.d. random variables from Rayleigh distribution, and Z^N are defined in the same way as in AWGN channel. If the receiver knows channel gains G^N, then it is the case of the ideal side information (SI), otherwise there is no side information.

We also consider an AWGN channel with bursty noise, which approximates interference in wireless channels:

$$Y^N = S^N + Z^N + W^N,$$

where $Z_i \sim \mathcal{N}(0, \sigma^2)$, and a component of bursty noise $W_i \sim \mathcal{N}(0, \sigma_b^2)$ with probability ρ and $W_i = 0$ with probability $1 - \rho$. The bursty noise parameter σ_b^2 is usually taken quite large to have noticeable influence on the performance.

The decoder receives Y^N and computes the log likelihood ratios (LLRs) L^n during demodulation. For each bit in a codeword L is computed as:

$$L = \log \frac{\Pr(X = 0|Y^N)}{\Pr(X = 1|Y^N)} \, .$$

Then the LLRs vector L^n is passed to the decoder which recovers a message $\hat{m} \in \{0, 1\}^k$.

Channel Coding

We limit our consideration of the hand-crafted codes to binary linear codes. The encoding process for a binary linear block code $\mathcal{C}$ can be given by multiplication over the binary field $\mathcal{F}$ by the binary generator matrix G of size $k \times n$: $X^n = mG$, $m \in \{0, 1\}^k$. The code rate is defined as $R = \frac{k}{n}$.

The aim of the decoding process is to recover the transmitted message m by exploiting the code structure. The hard-output decoders return a hard decision $\hat{m}$ that is an estimate of transmitted message m. Successive cancellation (SC) (Arikan 2008) and successive cancellation list (SCL) (Tal and Vardy 2011) for polar codes are hard-output decoders which show near optimal performance though have a low decoding throughput because of sequential decoding. Soft-output decoders return a soft estimate $\hat{X}^n \in \mathcal{R}^n$ instead of a hard decision $\hat{m}$. The sign of each position in the soft output $\hat{X}^n$ is an estimate of the codeword bit, and the hard decision $\hat{m}$ can be then obtained from the recovered codeword. The absolute values of $\hat{X}^n$ can be interpreted as a decoder's confidence for each bit. This is especially useful for decoding the concatenation of codes. One of the common examples of the soft-output decoder is an iterative BP algorithm which we describe further.

Commonly used metrics that measure decoding performance are the BLER (block error rate) defined as:

$$\text{BLER} = \Pr\left(\hat{m} \neq m\right), \tag{1}$$

and the BER (bit error rate) defined as:

$$\text{BER} = \frac{1}{k} \sum_{i=1}^{k} \Pr\left(\hat{m}_i \neq m_i\right). \tag{2}$$

BP Decoding

For decoding a binary linear block code $\mathcal{C}$ let the parity check matrix H of size $(n - k) \times n$ be such that $GH^T = 0$ over the binary field $\mathcal{F}$. Thus, each codeword $X^n \in \mathcal{C}$ satisfies $HX^n = 0$. BP is a message passing algorithm commonly used for decoding of linear block codes. BP decoding is performed over the Tanner graph that is a bipartite graphical representation of the parity check matrix H. BP iteratively exchanges messages between variable nodes (representing codeword symbols) and check nodes (representing parity constraints). As BP decoding performance for polar codes suffers from short cycles in the factor graph, recent research has improved BP decoding through factor graph permutations (Doan et al. 2018b; Elkelesh et al. 2018), and early stop aided by additional CRC (Ren et al. 2015).

Neural Decoding

Both model-free and model-based neural approaches for channel decoding are topics of active research. Adding trainable weights to node updates in BP decoding resulted in

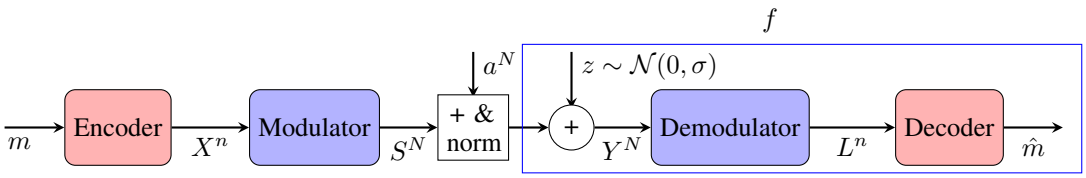

Figure 2: The communication scheme with a friendly attack a^N.

a popular model-based neural decoding called NBP decoding (Nachmani, Be'ery, and Burshtein 2016; Nachmani et al. 2018; Lugosch and Gross 2017). One of the main challenges in neural decoding remains the choice of the loss function. Binary cross-entropy (BCE) loss is often used between the codeword and its estimate:

$$\mathcal{L}(\mathbf{x}^n, \hat{\mathbf{x}}^n) = -\sum_{i=1}^{n} x_i \log \hat{x}_i + (1 - x_i) \log(1 - \hat{x}_i) \; . \quad (3)$$

Better results were obtained by averaging the BCE loss after each message passing iteration (Nachmani, Be'ery, and Burshtein 2016; Nachmani et al. 2018). However, reduction in the BCE loss does not necessarily result in BLER and BER, yet training the model directly using these metrics is not possible as they are not differentiable over the whole vector space $\mathbb{R}^n$. In the error correcting code transformer (Choukroun and Wolf 2022), which exhibited promising results for a wide range of linear block codes, authors compute a loss between binary multiplicative noise and model output aiming at good noise prediction for a given code structure. In (Kim et al. 2018), the authors aim at obtaining a performance similar to the BCJR algorithm known to be optimal for convolutional codes. Thus, they use a mean squared error (MSE) loss between the model output and the result of the conventional BCJR decoder for training.

Adversarial Attacks

Adversarial attacks aim at decreasing the neural network performance by introducing minor perturbations into its input. Channel decoding can be considered as a classification task with 2^k classes, where the decoder estimates the class of the transmitted message $m \in \{0, 1\}^k$. Let us denote here a neural network input as S, the corresponding true label as m and the loss function as $J(f(S), m)$, for example, the cross-entropy loss. A non-targeted adversarial attack is considered successful if the classifier misclassifies the adversarial sample: $f(S^{adv}) \neq m$, while the p-norm distance between the adversarial and original samples satisfies the constraint $||S^{adv} - S||_p < \epsilon$, as the goal is to make the attack as invisible to humans as possible. The optimization problem for generating an adversarial example is:

$$S^{adv} = \underset{||S^{adv} - S||_p < \epsilon}{\arg\max} J(f(S^{adv}), m) \; . \quad (4)$$

There are different attack methods which solve this optimization problem. The fast gradient sign method (FGSM) (Goodfellow, Shlens, and Szegedy 2015) solves (4) for $p = \infty$:

$$S^{adv} = S + \epsilon \cdot \text{sign}(\nabla_S J(f(S), m)) \; . \quad (5)$$

Algorithm 1: Search for Friendly Attack

Input: modulated codeword S^N, codeword or message m
Parameters: batch size B, number of iterations I, noise std σ, $gradient\_scheduler$
Output: $\hat{m}$

Let $i = 0$, $s_0 = S^N$ repeated B times, $y_{true} = m$ repeated B times, $a = 0$.
while $i < I$ **do**
  $\epsilon = gradient\_scheduler(i)$
  $(z_i)_{jk} \sim \mathcal{N}(0, \sigma), 1 \leq j \leq B, 1 \leq k \leq N$
  $y_i = s_i + z_i$,
  $l_i = demodulator(y_i)$ is a $B \times n$ matrix
  calculate BER, BLER for $decoder(l_i), y_{true}$
  $a_i = -\epsilon \nabla_{s_i} J(decoder(l_i), y_{true})$
  $\overline{a_i} = \frac{1}{B} \sum_{j=1}^{B} a_{ij}$ average over batch
  $\overline{a_i} = \overline{a_i}$ repeated B times
  $\hat{l_i} = demodulator(s_i + \overline{a_i} + z_i)$
  calculate $\hat{\text{BER}}$, $\hat{\text{BLER}}$ for $decoder(\hat{l_i}), y_{true}$
  if $\hat{\text{BER}} <$ BER and/or $\hat{\text{BLER}} <$ BLER **then**
    $i = i + 1$, $s_i = s_i + \overline{a_i}$, $a = a + \overline{a_i}$
  end if
end while
return a

The iterative version of FGSM (I-FGSM) (Kurakin, Goodfellow, and Bengio 2016) applies the gradient update in (5) multiple times with some rate α:

$$S_0^{adv} = S, \; S_t^{adv} = S_{t-1}^{adv} + \alpha \, \text{sign}(\nabla_S J(f(S_{t-1}^{adv}), m)). \quad (6)$$

Here, we can set $\alpha = \frac{\epsilon}{T}$, where T is the number of iterations, for consistency with (4). It has been shown that iterative methods achieve better results in terms of successful attacks (Kurakin, Goodfellow, and Bengio 2017). The attack in (5), (6) can also be generalised to other p-norms.

Friendly Attack Against a Channel Decoder

In this work, we suggest a new concept called a 'friendly attack' on channel decoders to improve their reliability.

Modified Scheme

As previously mentioned, the decoder can be considered as a classifier with 2^k classes as it recovers a message $\hat{m} \in \{0, 1\}^k$. The idea is to use the gradient method to find a perturbation vector for each codeword that will improve the decoding performance, rather than attacking it. In our scheme, we suggest to add the perturbation vector a^N to the modulated

codeword S^N before transmission over the channel as shown in Fig. 2. The aim is to find such a vector a^N that without violation of the average power constraint the decoding error (BLER or BER) on average over the channel distribution will be smaller. So the model f under attack is not only decoder itself but also the noisy channel and demodulator as illustrated with a blue box in Fig. 2.

The differences between a conventional adversarial setting and our formulation are as follows:

- As we want to increase the performance of the decoder, the optimization problem should be modified to:
$$S^{frnd} = \arg\min_{S^{frnd}} J(f(S^{frnd}), m) \, . \tag{7}$$
It can be considered as a similar formulation to the targeted attack (Kurakin, Goodfellow, and Bengio 2016), but here the target class is the true one.
- In the optimisation problem in (7) there is no constraint on a p-norm distance between the codeword and the attacked version $S^{frnd} = S^N + a^N$, i.e., there is no constraint on the p-norm of vector a^N. Nevertheless, the average power constraint has to be preserved for an attacked codeword $S^N + a^N$. This can be achieved by applying an additional normalisation before transmitting the codeword:
$$||C(S^N + a^N)||_2^2 \leq NP \, ,$$
where $C = \frac{\sqrt{NP}}{||S^N + a^N||_2}$ is the normalisation constant. Thus, we can rewrite (6) into an iterative gradient attack:
$$S_0^{frnd} = S, \; S_t^{frnd} = S_{t-1}^{frnd} - \epsilon \nabla_S J(f(S_{t-1}^{frnd}), m) \, , \tag{8}$$
or in terms of the attack vector:
$$a_0^N = 0, \; a_t^N = a_{t-1}^N - \epsilon \cdot \nabla_S J(f(S + a_{t-1}^N), m) \, . \tag{9}$$
- The model we intend to attack is stochastic due to the presence of channel noise, which constitutes the primary challenge in our task. We want the attack vector a^N for the modulated codeword S^N to be good on average for noisy transmission in terms of BLER and BER.

In general, any given decoder splits the channel output space into 2^k decision regions and the input of the decoder is a vector of LLRs L^n from one of these regions. The modulated codewords then might be not optimal for the fixed decoder as they may not correspond to the center of the decision regions. Thus, a small shift of the modulated codeword in the direction of the decision region's center should increase the decoding performance. We will say that our friendly attack is successful if it improves the decoding performance on average for a given codeword. As we consider linear codes here, the successful attack vector for one codeword can be easily applied to all codewords (with appropriate sign adaptation) due to the code linearity and must show similar improvements. Let us suppose that we have found an attack a^N for an all-zero codeword for some modulation with a constellation point corresponding to all zero bits c_0 ($c_0 \in \mathrm{R}$ for real constellations, $c_0 \in \mathrm{C}$ for complex constellations), then the attacked codeword will be:
$$S^{frnd} = S^N + \frac{S^N a^N}{S_0} \, , \tag{10}$$
where normalised attack vector $\frac{S^N a^N}{S_0} = \frac{1}{S_0}\{S_i a_i\}_{i=1}^N$.

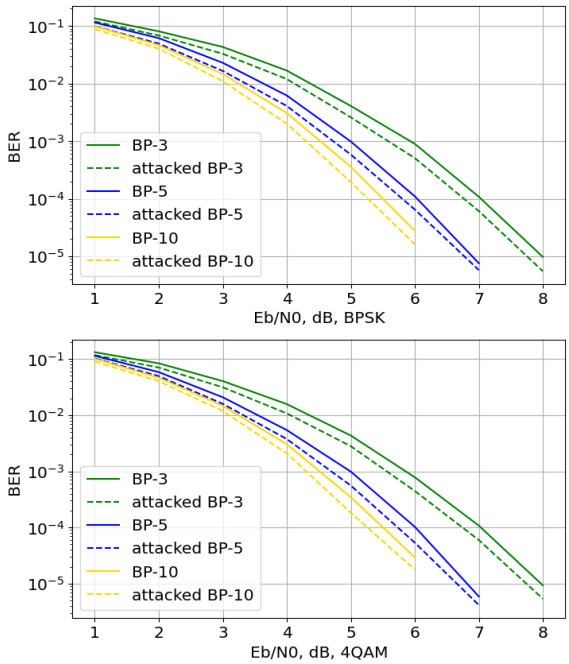

Figure 3: BER for LDPC code $(64, 32)$ with and without attack using BPSK (top) and 4-QAM (bottom) modulations. 'BP-n' refers to BP decoder with n iterations.

Attack

To design an attack vector a^N we need to be able to compute $\nabla_S J(f(S), m)$. The model f consists of the noisy channel, the demodulator and the channel decoder as in Fig. 2. AWGN channel and the demodulator are differentiable. We consider BP decoder as it is differentiable, as well as NBP and other neural decoders such as ECCT (Choukroun and Wolf 2022) for linear codes.

To overcome the problem with stochasticity of the transmission we apply a gradient update (9) over a batch of size B as in Alg. 1. That means we send the same codeword S^N over the channel B times for every iteration (line 3 in Alg. 1). In this section and in Alg. 1, we use $\nabla_S J(f(S), m)$ as a gradient from the decoder for the batch of codewords S^N. All B attack vectors are then averaged over the batch and added to the modulated codeword to check (line 11) if they indeed improve the decoding performance. This is not necessary for all architectures we tried, but crucial for architectures like ECCT as it aims at noise prediction, and the gradient direction obtained for this decoder might be bad on average over the batch.

The choice of parameters I, B, and a scheduler for ϵ defines an attack. We can highlight four different approaches:

1. In the first approach, we employ a rather large batch size $B \sim 1000 - 10000$ and relatively small number of iterations $I \sim 1 - 100$.
2. The second approach needs an average batch size $B \sim 100 - 500$ and relatively high number of iterations $I \sim 500 - 5000$. For this setting a scheduler for ϵ should be chosen carefully. If this approach is applicable, it gives the

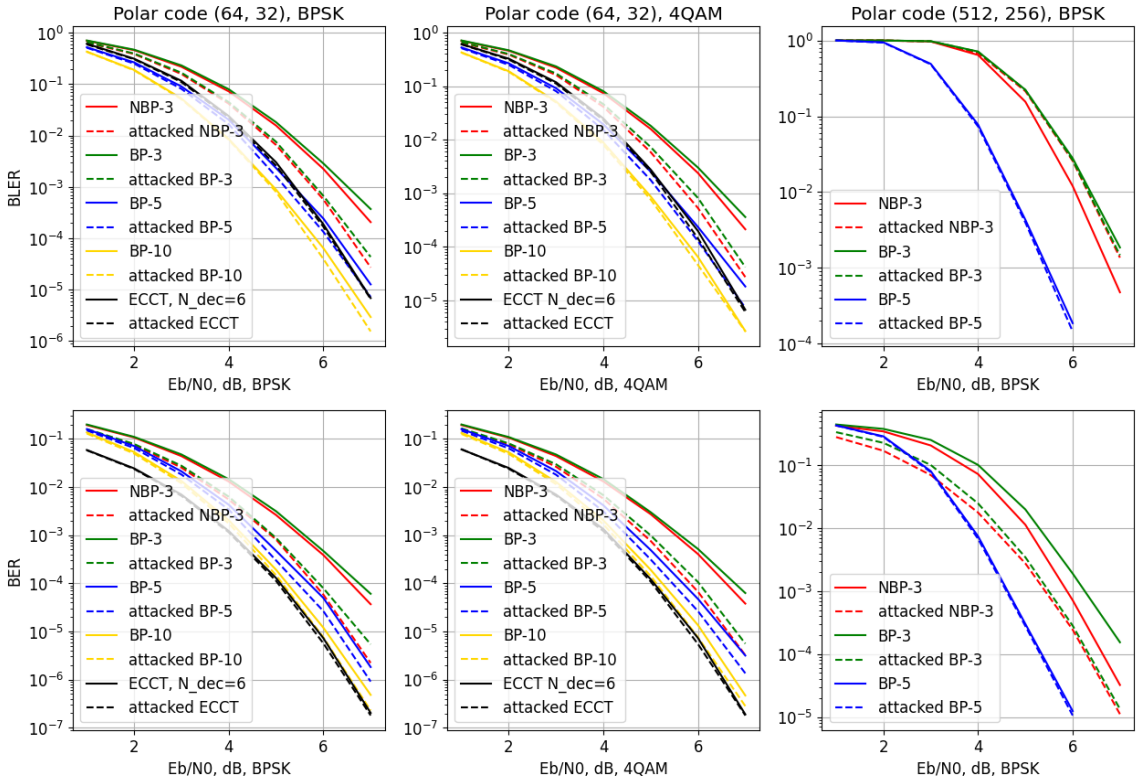

Figure 4: BLER (top) and BER (bottom) for $(64, 32)$ and $(512, 256)$ polar codes with BPSK and 4-QAM modulations.

best overall results. However, it does not perform well for some decoders and modulations due to a big bias towards the noise in a batch.

3. The third and fourth approaches are more robust as they need multiple runs of Alg. 1 and saving output attack vectors after each run. The final attack vector is found by running a clustering algorithm on the set of attack vectors from multiple runs and then choosing the best one. The best results were obtained with Agglomerative Clustering (with linkage criterion "ward" and "complete") and K-means (with number of neighbours $3 - 4$). For both approaches, the scikit-learn library implementation (Pedregosa et al. 2011) was used. Finally, the third approach includes a relatively small batch size $B \sim 5 - 50$, number of iterations $I \sim 15 - 40$ and many runs of algorithm 1: $1000 - 5000$.
4. The fourth approach is similar to the first one but is run multiple times with a small number of iterations $I \sim 1 - 5$. This method is time consuming and works for neural models which were trained on the input ± 1 with little noise introduced as they are vulnerable to any perturbation of the input.

Such a variety of approaches in attacks and their different results in different settings (code, decoder type, modulation) can be explained by the different splits of the codeword space. If there is more than one direction for the gradient descent the algorithm may struggle, and the third and fourth approaches work the best because final clustering allows to have more than one optimal perturbation vector.

Taking the gradient $\nabla_S J(f(S), m)$ during decoding and adding it to a codeword can be interpreted as greater power allocation for symbol positions in S^N which have more impact on the error correction. The decoder corrects errors only if there is some noise introduced during transmission otherwise there is no error to correct. It means that it is possible to find a successful attack vector only during noisy transmission. If an attack vector is obtained in a high signal-to-noise ratio (SNR) regime (almost no noise), the decoder has almost no errors to correct, and it is difficult to identify an effective attack vector. If an attack vector is obtained at a low SNR (SNRs for which $\text{BLER} \geq 0.7$), there is also almost no error correction because the decoder is not able to extract any information from the noisy channel output. We saw in our experiments that the best attack vectors are found in SNR regimes where $\text{BLER} \sim 0.1 - 0.5$.

Simulation Results

In this work, we do not aim at proposing a new efficient decoding approach, but to show that our friendly attack can improve the performance of the variety of existing codes and decoding algorithms in different scenarios. To compare our results with the state-of-the-art approaches we used the sionna library (Hoydis et al. 2022), which follows the 5G standard (3GPP 2018). We consider both polar and LDPC codes with different decoding approaches and modulations. LDPC code design in our experiments is the same as in

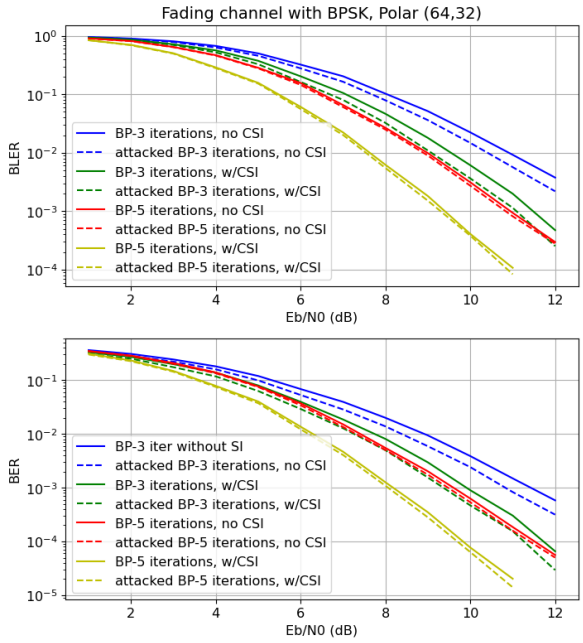

Figure 5: BLER (top) and BER (bottom) for $(64, 32)$ polar code over a fading channel with BPSK.

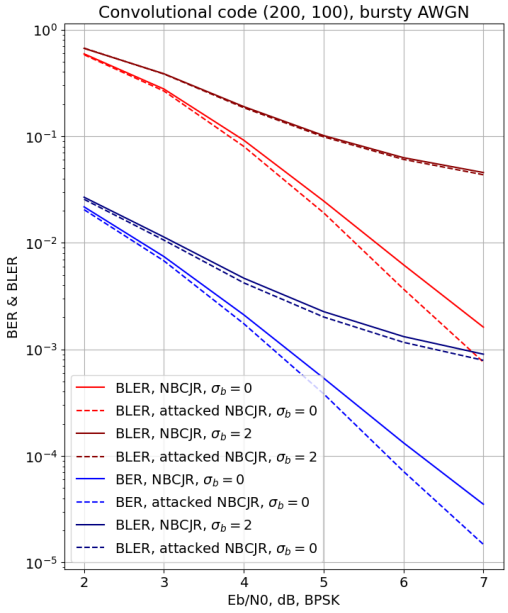

Figure 6: BLER and BER for the $(200, 100)$ convolutional code with BPSK modulation and NBCJR decoding.

the sionna library. We are particularly interested in short block lengths as near capacity performance can be achieved by LDPC codes in large block lengths. All attacks were obtained using the all-zero codeword, and then tested on random codewords. All the simulations were run on Nvidia RTX A6000 GPUs with 48GB of memory.

We first consider the $(64, 32)$ LDPC code with BP decoding (MacKay and Neal 1997) for different number of iterations and different modulations. BER of BP decoding with 3 and 5 iterations and their best attacks are presented in Fig. 3 for BPSK and 4-QAM modulations. Improvement for BLER performance for these decoders via friendly attacks has similar behaviour and is omitted here. We consider here relatively small number of iterations, since with the increasing number of iterations, the BP decoder gets closer to the optimal decoding, and is harder to improve.

For comparison we considered also polar codes: $(64, 32)$ for a short block length and $(512, 256)$ for a long block length. We consider both BP and NBP decoding for polar codes. The results are presented in Fig. 4. For long block length, $k = 256$, there is no improvement in BLER for NBP while BER performance was noticeably enhanced via the friendly attack for both BP and NBP decoders. It can be noticed that improvement in BER for BP decoding is greater than that for NBP. We also include the ECCT (Choukroun and Wolf 2022) decoding results with $N_{dec} = 6$ for code $(64, 32)$ and its attacks for both modulations. Although the improvement via attack for ECCT is barely visible, it shows around $\%10$ reduction in BER and BLER for > 4 dB even for this complex and deep model.

In Fig. 5, we present BLER and BER results for polar code $(64, 32)$ over a Rayleigh fading channel with BPSK modulation. For this scenario, we considered BP decoding with 3 and 5 iterations both with and without side information (SI). Similarly to previous results, more significant improvement can be achieved for the suboptimal BP decoder with 3 iterations.

As an additional evidence of the adaptability of the friendly attack method we also provide the results for NBCJR (Kim et al. 2018) for convolutional code of rate $R = \frac{1}{2}$ and block length $k = 100$ in Fig. 6 for both AWGN and bursty AWGN channels.

Conclusion

In this work, we proposed a new concept of a 'friendly attack' for channel coding that employs the adversarial attack approach to enhance the channel decoding. The friendly attack is a vector found by a proposed here iterative algorithm based on a gradient descent during transmission and decoding. We showed that the addition of the attack vector to a modulated codeword before transmission significantly improves the error correction performance of the decoder without any changes on the decoder's side. The proposed approach was tested for a range of codes and decoders, and also for different modulations and for different channels. Proposed friendly attack showed promising results in terms of BER for different decoders and it appeared to be more efficient in BER improvement for BP decoding compared to NBP for a small number of decoding iterations.

Acknowledgements

This work received funding from the UKRI for the project AIR (European Resarch Council Consolidator Grant, EP/X030806/1).

References

3GPP. 2018. Multiplexing and channel coding (Release 10) 3GPP TS 21.101 v10.4.0. https://www.3gpp.org/ftp/Specs/2023-06/Rel-10/21_series/21101-a40.zip. Accessed: 2023-08-13.

Aref, V.; and Chagnon, M. 2022. End-to-End Learning of Joint Geometric and Probabilistic Constellation Shaping. In *2022 Optical Fiber Communications Conference and Exhibition (OFC)*, 1–3.

Arikan, E. 2008. Channel polarization: A method for constructing capacity-achieving codes. In *2008 IEEE International Symposium on Information Theory*, 1173–1177.

Bennatan, A.; Choukroun, Y.; and Kisilev, P. 2018. Deep Learning for Decoding of Linear Codes - A Syndrome-Based Approach.

Berrou, C.; Glavieux, A.; and Thitimajshima, P. 1993. Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1. In *Proceedings of ICC '93 - IEEE International Conference on Communications*, volume 2, 1064–1070 vol.2.

Buchberger, A.; Häger, C.; Pfister, H.; Schmalen, L.; and Graell i Amat, A. 2020. Learned Decimation for Neural Belief Propagation Decoders.

Choukroun, Y.; and Wolf, L. 2022. Error Correction Code Transformer. arXiv:2203.14966.

Doan, N.; Ali Hashemi, S.; and Gross, W. J. 2018. Neural Successive Cancellation Decoding of Polar Codes. In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 1–5.

Doan, N.; Hashemi, S. A.; Mambou, E. N.; Tonnellier, T.; and Gross, W. J. 2018a. Neural Belief Propagation Decoding of CRC-Polar Concatenated Codes. *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 1–6.

Doan, N.; Hashemi, S. A.; Mondelli, M.; and Gross, W. 2018b. On the Decoding of Polar Codes on Permuted Factor Graphs. 1–6.

Elias, P. 1955. Coding for noisy channels. *IRE Conv. Rec.*, 3: 37–46.

Elkelesh, A.; Ebada, M.; Cammerer, S.; and ten Brink, S. 2018. Belief propagation decoding of polar codes on permuted factor graphs. In *2018 IEEE Wireless Communications and Networking Conference (WCNC)*, 1–6.

Gallager, R. 1962. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1): 21–28.

Goodfellow, I. J.; Shlens, J.; and Szegedy, C. 2015. Explaining and Harnessing Adversarial Examples. In Bengio, Y.; and LeCun, Y., eds., *International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Gruber, T.; Cammerer, S.; Hoydis, J.; and Brink, S. t. 2017. On deep learning-based channel decoding. In *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, 1–6.

Gümüş, K.; Alvarado, A.; Chen, B.; Häger, C.; and Agrell, E. 2020. End-to-End Learning of Geometrical Shaping Maximizing Generalized Mutual Information. In *2020 Optical Fiber Communications Conference and Exhibition (OFC)*, 1–3.

Hameed, M. Z.; György, A.; and Gündüz, D. 2021. The Best Defense Is a Good Offense: Adversarial Attacks to Avoid Modulation Detection. *IEEE Transactions on Information Forensics and Security*, 16: 1074–1087.

Hou, J.; Siegel, P.; and Milstein, L. 2001. Performance analysis and code optimization of low density parity-check codes on Rayleigh fading channels. *IEEE Journal on Selected Areas in Communications*, 19(5): 924–934.

Hoydis, J.; Cammerer, S.; Ait Aoudia, F.; Vem, A.; Binder, N.; Marcus, G.; and Keller, A. 2022. Sionna: An Open-Source Library for Next-Generation Physical Layer Research. *arXiv preprint*.

Jones, R. T.; Yankov, M. P.; and Zibar, D. 2019. End-to-end learning for GMI optimized geometric constellation shape. In *45th European Conference on Optical Communication (ECOC 2019)*, 1–4.

Kim, H.; Jiang, Y.; Rana, R. B.; Kannan, S.; Oh, S.; and Viswanath, P. 2018. Communication Algorithms via Deep Learning. In *International Conference on Learning Representations*.

Kurakin, A.; Goodfellow, I.; and Bengio, S. 2016. Adversarial examples in the physical world.

Kurakin, A.; Goodfellow, I. J.; and Bengio, S. 2017. Adversarial Machine Learning at Scale. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net.

Lugosch, L.; and Gross, W. J. 2017. Neural Offset Min-Sum Decoding. In *2017 IEEE International Symposium on Information Theory (ISIT)*, 1361–1365. IEEE Press.

MacKay, D. J.; and Neal, R. M. 1997. Near Shannon limit performance of low density parity check codes. volume 33, 457–458.

Makkuva, A. V.; Liu, X.; Jamali, M. V.; Mahdavifar, H.; Oh, S.; and Viswanath, P. 2021. KO codes: inventing nonlinear encoding and decoding for reliable wireless communication via deep-learning. In Meila, M.; and Zhang, T., eds., *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, 7368–7378. PMLR.

Nachmani, E.; Be'ery, Y.; and Burshtein, D. 2016. Learning to decode linear codes using deep learning. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 341–346.

Nachmani, E.; Marciano, E.; Lugosch, L.; Gross, W. J.; Burshtein, D.; and Be'ery, Y. 2018. Deep Learning Methods for Improved Decoding of Linear Codes. *IEEE Journal of Selected Topics in Signal Processing*, 12(1): 119–131.

O'Shea, T.; and Hoydis, J. 2017. An Introduction to Machine Learning Communications Systems. *ArXiv*, abs/1702.00832.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.;

Brucher, M.; Perrot, M.; and Duchesnay, E. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12: 2825–2830.

Polyanskiy, Y.; Poor, H. V.; and Verdú, S. 2010. Channel Coding Rate in the Finite Blocklength Regime. 56(5): 2307–2359.

Qu, Z.; and Djordjevic, I. B. 2019. On the Probabilistic Shaping and Geometric Shaping in Optical Communication Systems. *IEEE Access*, 7: 21454–21464.

Ren, Y.; Zhang, C.; Liu, X.; and You, X. 2015. Efficient early termination schemes for belief-propagation decoding of polar codes. In *2015 IEEE 11th International Conference on ASIC (ASICON)*, 1–4.

Shannon, C. E. 1948. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3): 379–423.

Tal, I.; and Vardy, A. 2011. List decoding of polar codes. In *2011 IEEE International Symposium on Information Theory Proceedings*, 1–5.

Xu, W.; Wu, Z.; Ueng, Y.-L.; You, X.; and Zhang, C. 2017. Improved polar decoder based on deep learning. In *2017 IEEE International Workshop on Signal Processing Systems (SiPS)*, 1–6.