Revista Brasileira de Computação Aplicada, Novembro, 2022

DOI: 10.5335/rbca.v14i3.13278
Vol. 14, Nº 3, pp. 86–95
Homepage: seer.upf.br/index.php/rbca/index

ARTIGO ORIGINAL

Otimização de um decodificador neural para códigos BCH curtos sob regime de comunicação crítica

Optimization of a neural decoder for short BCH codes under critical communication regime

Jorge Kysnney Santos Kamassury [1]

[1]Universidade Federal de Santa Catarina (UFSC)

*jorge.kamassury@posgrad.ufsc.br

Recebido: 29/12/2021. **Revisado:** 02/11/2022. **Aceito:** 15/11/2022.

Resumo

No presente trabalho, introduz-se no contexto dos códigos corretores de erros uma estratégia de decodificação onde uma rede neural é treinada para predizer padrões de erros usando simultaneamente as informações dos módulos e das síndromes dos vetores recebidos. No decodificador proposto, as posições mais confiáveis são selecionadas de forma iterativa para serem os bits errôneos do padrão de erro estimado, de modo que estas são posteriormente subtraídas do vetor recebido antes que uma nova decodificação seja realizada. Para a predição do padrão de erro, projeta-se uma rede neural profunda com complexidade reduzida. Os experimentos realizados para os códigos BCH curtos transmitindo via canal AWGN evidenciam que os desempenhos obtidos com essa estratégia de decodificação superam àqueles obtidos exclusivamente com a rede neural.

Palavras-Chave: Códigos BCH; códigos corretores de erros; códigos de comprimento curto; decodificador iterativo; vetor síndrome.

Abstract

In the present paper, a decoding strategy is introduced in the context of error-correcting codes where a neural network is trained to predict error patterns using simultaneously the information from the modules and the syndromes of the received vectors. In the proposed decoder, the most reliable positions are iteratively selected to be the erroneous bits of the estimated error pattern, so these are later subtracted from the received vector before a new decoding is performed. For the prediction of the error pattern, a deep neural network with reduced complexity is designed. The experiments carried out for short BCH codes transmitting via the AWGN channel show that the performances obtained with this decoding strategy surpass those obtained exclusively with the neural network.

Keywords: BCH codes; error correcting codes; short length codes; iterative decoder; syndrome vector.

1 Introdução

Nos últimos anos, estudos relacionados aos projetos de códigos de comprimento curto têm sido potencializados, especialmente, em virtude dos serviços que tecnologias mais recentes objetivam dar suporte.

Hoje, muitas aplicações visam, por exemplo, comunicações orientadas a humanos e para tanto são projetadas/dimensionadas para assegurar boa conectividade na maior parte do tempo; contudo, a taxa de transmissão é quase nula em cenários nos quais há parca cobertura, sobrecarga

ou interferência significativa. Por outro lado, sistemas emergentes, com comunicação orientada a máquinas, dependem criticamente da disponibilidade quase ininterrupta dos enlaces de rádio, inclusive, com qualidade mínima de comunicação. Além disso, sabe-se que os sistemas tradicionais de comunicação orientada a humanos são caracterizados por pacotes onde a quantidade de dados transmitidos é consideravelmente maior que a informação de controle correspondente. Entretanto, a maioria do tráfego promovido pela comunicação entre máquinas é constituída de pacotes curtos que, em geral, carregam informação crítica que demandam uma baixa latência e confiabilidade elevada (Cavalcanti et al., 2018). Infelizmente, a efetivação de uma comunicação governada por esses requisitos (alta confiabilidade e baixa latência) não é uma tarefa trivial, pois os próprios requisitos são altamente estritos e conflitantes (Bennis et al., 2018).

Neste cenário, os atuais estudos no âmbito dos códigos corretores de erros (*error correcting codes*, ECCs) têm evidenciado que a abordagem clássica que faz uso de códigos de comprimento muito longos é inadequada devido ao requisito de latência. Isto posto, as pesquisas têm sido direcionadas para avaliar possíveis códigos curtos candidatos em relação a parâmetros como comprimento ($k < 1000$ bits), confiabilidade, latência e complexidade algorítmica. No que se refere à confiabilidade, os códigos BCH[1] superam todos os outros códigos (LDPC, convolucionais, etc.) devido às grandes distâncias mínimas[2] que são características desse tipo de código. Em relação à taxa, os códigos BCH também superam os demais candidatos com um desempenho muito próximo ao *benchmark* da aproximação normal (Shirvanimoghaddam et al., 2019). Todavia, conforme acentua Kamassury (2021), um dos grandes desafios enfrentados pela classe de códigos BCH está relacionado ao fato de que sua decodificação usando algoritmos tradicionais é significativamente complexa.

Uma alternativa relevante aos algoritmos tradicionais de decodificação consiste no uso de redes neurais profundas (*deep neural networks*, DNNs). Por exemplo, Nachmani et al. (2016) propuseram um decodificador baseado em redes neurais (*neural networks*, NNs) para melhorar o desempenho do algoritmo *belief propagation* (BP) para decodificar códigos BCH de comprimento curto. Na abordagem proposta, o algoritmo BP com l iterações é convertido em $2l$ camadas ocultas em uma rede densa. Em trabalhos mais recentes e baseados nesta abordagem apresentada por Nachmani et al. (2016), Lugosch and Gross (2018) introduziram uma nova função de perda baseada na relaxação do vetor síndrome, penalizando as estimativas da rede neural que não correspondem às palavras-código. Por sua vez, usando a técnica de aprendizado ativo, Be'ery et al. (2020) orientam seus estudos para a amostragem inteligente do conjunto de treinamento.

Na prática, uma desvantagem da abordagem que busca reproduzir a estrutura do decodificador BP é que as conexões são reguladas para se assemelhar ao gráfico de Tanner, o que limita consideravelmente a flexibilidade para uso de outras arquiteturas neurais.

Diferentemente da abordagem baseado no decodificador BP, Bennatan et al. (2018) propuseram uma estrutura de decodificação, onde a NN é alimentada pelos módulos (confiabilidades) e os vetores síndromes das sequências recebidas e atua na estimação do ruído do canal de transmissão. De fato, a grande vantagem dessa estrutura é a sua flexibilidade, permitindo que a rede possa ser projetada livremente

Isto posto, no presente trabalho apresenta-se uma nova estratégia para otimizar a performance do decodificador proposto por Bennatan et al. (2018) ao custo de um pequeno incremento de complexidade computacional. Na abordagem proposta, seleciona-se iterativamente as posições mais confiáveis para serem os bits errôneos do padrão de erro, atualizando o vetor recebido quando uma nova posição do erro padrão é selecionada. Para a estimativa do padrão de erro, projeta-se uma nova NN com complexidade reduzida.

Os resultados para os códigos BCH investigados evidenciam que esta estratégia (decodificador iterativo) otimiza o desempenho de decodificação independentemente da métrica utilizada.

2 Modelo de um sistema de comunicação e códigos de bloco

Seja o modelo de comunicação digital esquematizado na Fig. 1 Neste esquema, uma sequência de dígitos $q-$ária produzida por uma fonte é subdividida em blocos de k dígitos cada, denotados pelo vetor $\mathbf{u}$ que alimentam o codificador. Em seguida, o codificador adiciona redundância ao vetor $\mathbf{u}$, produzindo um vetor $\mathbf{c}$ (com n dígitos) chamado de palavra-código, onde $n \geq k$. Ao conjunto de todas as palavras-código atribui o nome de *código de bloco*.

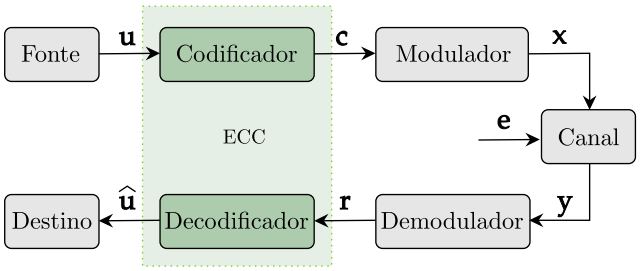

Figura 1: Esquema de um sistema de comunicação digital codificado

Ainda com base na Fig. 1, o bloco modulador converte o vetor $\mathbf{c}$ em um vetor de sinal modulado $\mathbf{x}$ adequado para transmissão em canais analógicos que estão sujeitos às

[1] Descobertos por Bose, Chaudhuri e Hocquenghem, os códigos BCH representam uma das classes mais importantes e poderosas de códigos de bloco lineares (Hocquenghem, 1959; Bose and Ray-Chaudhuri, 1960b,a). Com capacidade de correção de t erros, esses códigos podem detectar e corrigir múltiplos erros, sendo assim considerados uma generalização dos códigos de Hamming. Além disso, possuem flexibilidade na escolha de parâmetros como comprimento e taxa de código.

[2] Em teoria da informação, a distância mínima é a menor distância de Hamming entre duas palavras códigos válidas de um código, enquanto que a distância de Hamming é compreendida como o número de posições em que dois vetores diferem.

variadas fontes de erros e que podem corromper a entrada do canal $\mathbf{x}$, tornando a saída $\mathbf{y}$ diferente da informação modulada transmitida (isto é, $\mathbf{x}$).

Por fim, do lado receptor, o bloco modulador aplicando alguma operação inversa produz uma saída $\mathbf{r}$. Neste caso, usando a redundância presente no vetor $\mathbf{c}$, o decodificador atuará corrigindo possíveis erros no vetor recebido $\mathbf{y}$, e dessa forma, obter uma estimativa de $\hat{\mathbf{u}}$.

De modo geral, o projeto de codificação/detecção/correção de erros deve lidar com três aspectos fundamentais:

- Criação de um código com propriedades de redundância desejadas;
- Estabelecer um algoritmo de decodificação para que a informação possa ser reproduzida, com determinada confiabilidade, na saída do decodificador;
- Tornar o sistema de codificação o mais eficiente possível, de modo que seja transmitida uma quantidade mínima de informações redundantes.

Para uma melhor compreensão dos detalhamentos das seções seguintes, utiliza-se um esquema simplificado do sistema de comunicação da Fig. 1, onde os blocos modulador, canal e demodulador estão combinados em único bloco denominado canal de codificação (vide Fig. 2). Neste caso, a entrada e a saída deste bloco são denotadas por $\mathbf{c}$ e $\mathbf{r}$, respectivamente (Lin and Costello Jr, 2004).

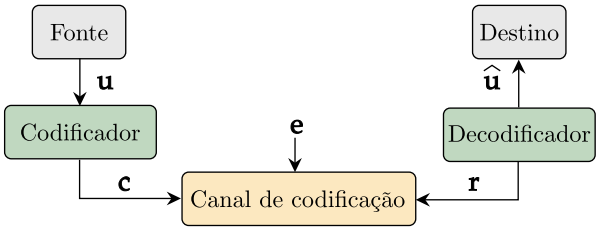

Figura 2: Modelo simplificado de um sistema de comunicação digital codificado

2.1 Descrição matricial dos códigos de bloco

Por definição, um código de bloco de comprimento n e 2^k palavras-código é dito um código linear (n, k) se, e somente se, suas 2^k palavras-código formam um subespaço $k-$dimensional do espaço vetorial de todas as $n-$tuplas sobre GF(2). A partir de k palavras-código linearmente independentes, pode-se obter qualquer palavra-código usando uma combinação linear das k palavras-código da base (Kamassury et al., 2019). Em geral, essas k palavras-código podem ser obtidas usando uma matriz geradora denotada por $\mathbf{G}$, tal que

$$\mathbf{c} = u_0\mathbf{G}_0 + u_1\mathbf{G}_1 + \cdots + u_{k-1}\mathbf{G}_{k-1} = \mathbf{uG} \tag{1}$$

onde $\mathbf{G}_{k\times n} = \left[\mathbf{P}_{(k\times n-k)}|\mathbf{I}_k\right]$, em que $\mathbf{I}$ é a *matriz identidade* e $\mathbf{P}$ é a *matriz de paridade* que corresponde aos dígitos utilizados para detecção e/ou correção de erro(s).

Há ainda a existência de uma matriz $\mathbf{H}$ com dimensão $(n-k)\times n$ cujas $n-k$ linhas são linearmente independentes, de modo que

$$\mathbf{GH}^t = 0\,, \quad \mathbf{cH^t} = \mathbf{0} \tag{2}$$

em que $\mathbf{H}_{(n-k\times n)} = \left[\mathbf{I}_{(n-k)}|\mathbf{P}^t\right]$ é denominada de *matriz de verificação de paridade*. Esta última expressão indica que qualquer palavra-código $\mathbf{c}$ produzida por $\mathbf{G}$ é ortogonal às linhas de $\mathbf{H}$; em outras palavras, qualquer n-upla $\mathbf{c}$ é uma palavra-código do código $\mathcal{C}$ se, e somente se, respeitar a Eq. (1).

2.2 Síndrome e suas propriedades

Seja uma palavra-código $\mathbf{c}$ transmitida por um canal ruidoso em que o vetor recebido $\mathbf{r}$ é expresso por

$$\mathbf{r} = \mathbf{c} + \mathbf{e} \tag{3}$$

onde $\mathbf{e}$ denota o *padrão de erro* (também conhecido como *vetor de erro*). Na prática, algoritmos de decodificação iniciam computando o vetor síndrome ($\mathbf{s}$) que é definido por:

$$\mathbf{s} = \mathbf{rH^t} \tag{4}$$

Haykin (2014) acentua que a síndrome possui duas importantes propriedades, a saber:

- Propriedade 1: o vetor síndrome depende exclusivamente do padrão de erro ($\mathbf{e}$) e não da palavra-código transmitida, ou seja $\mathbf{s} = \mathbf{eH^t}$;
- Propriedade 2: todos os padrões de erro que diferem por uma palavra-código têm a mesma síndrome.

2.3 Decodificação via vetor síndrome

Seja o vetor recebido $\mathbf{c}$ com 2^n valores possíveis, um decodificador atua particionando esse conjunto em 2^k subconjuntos disjuntos $\Omega_1, \cdots, \Omega_{2^k}$ em que Ω_i corresponde a $\mathbf{c}_i$ para $1 \leq i \leq 2^k$. Neste sentido, para que a decodificação tenha sucesso, $\mathbf{r}$ deve estar no subconjunto pertencente a $\mathbf{c}_i$ que foi enviada. Aos conjuntos 2^k atribui-se o nome de *arranjo padrão* de um código de bloco linear.

Na prática, um arranjo padrão é uma matriz na qual as 2^k colunas representam os subconjuntos disjuntos, enquanto as 2^{n-k} linhas são identificadas como *cosets* do código, cujos primeiros elementos são denominados de *líderes de cosets*. Lin and Costello Jr (2004) descrevem que estes líderes são os vetores com os menores pesos de Hamming[3].

Isto posto, o processo de decodificação usando o vetor síndrome possui as seguintes etapas:

- Computa-se o vetor síndrome: $\mathbf{s} = \mathbf{rH}^t$;
- Caso $\mathbf{s} = 0$, admite-se que $\mathbf{r}$ não contém erro. Caso contrário, busca-se o coset caracterizado por $\mathbf{s}$ (visu-

[3]Número de elementos não-nulos de um vetor.

alizando o arranjo padrão), identificando o coset líder, denotando-o por $\mathbf{e}_0$;
- Estima-se $\mathbf{c} = \mathbf{r} + \mathbf{e}_0$ como a decodificação de $\mathbf{r}$.

3 Redes neurais profundas

As redes neurais profundas consistem em aproximações de funções constituídas por uma série de camadas, em que cada camada representa alguma transformação de ativações de entrada para saída a partir de uma função de transferência paramétrica com conjuntos de pesos aprendidos (O'Shea and Hoydis, 2017).

Especificamente, uma NN do tipo perceptron com L camadas, representa um mapeamento $f(\mathbf{r}_0;\theta) : \mathcal{R}^{N_0} \rightarrow \mathcal{R}^{N_L}$ de um vetor de entrada $\mathbf{r}_0 \in \mathcal{R}^{N_0}$ para um vetor de saída $\mathbf{r}_L \in \mathcal{R}^{N_L}$ mediante L etapas, tal que:

$$\mathbf{r}_l = f_l(\mathbf{r}_{l-1};\theta_l), \quad l = 1, \cdots, L \tag{5}$$

Para o caso de uma arquitetura com camadas totalmente conectadas (vide Fig. 3), a l–ésima camada tem o mapeamento denotado por:

$$f_l(\mathbf{r}_{l-1};\theta_l) = g(\mathbf{W}_l\mathbf{r}_{l-1} + \mathbf{b}_l) \tag{6}$$

onde:

- $\mathbf{W}_l \in \mathcal{R}^{N_l \times N_{l-1}}$ é a matriz de pesos da camada l;
- $\mathbf{b}_l \in \mathcal{R}^{N_L}$ denota o vetor de *bias*;
- $g(\cdot)$ representa a função de ativação, em geral, aplicada individualmente a cada elemento de entrada da camada, isto é, $[g(\mathbf{z})]_i = g(z)_i$;
- $\theta_l = \{\mathbf{W}_l, \mathbf{b}_l\}$ é o conjunto de parâmetros da camada l.

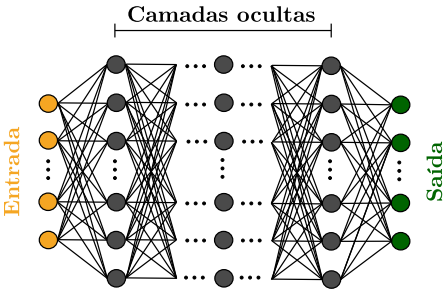

Figura 3: Rede neural profunda

A principal característica de uma NN densa reside no fato de que cada neurônio recebe na entrada a contribuição de cada neurônio da camada anterior e alimenta sua saída para todos os neurônios da próxima camada.

No âmbito do aprendizado de máquina, reconhece-se que o potencial das redes neurais está intrínseco a sua configurabilidade. Em geral, a arquitetura neural é configurada por um procedimento de treinamento que depende de um conjunto de dados de amostra, consistindo de entradas e, no caso de treinamento supervisionado, de saídas desejadas (*rótulos*). Desse modo, o processo de treinamento é realizado usando dados rotulados, isto é, um conjunto de pares de vetores entrada-saída $\left(\mathbf{r}_{0,j}, \mathbf{r}^{\dagger}_{L,j}\right)$ para $j \in [1, \cdots, N]$, em que $\mathbf{r}^{\dagger}_{L,j}$ é o valor da saída desejada.

Com efeito, o propósito do treinamento incorre em minimizar a função de custo

$$\mathcal{J}(\theta) = \frac{1}{N}\sum_{j=1}^{N} \mathcal{L}\left(\mathbf{r}^{\dagger}_{L,j}, \mathbf{r}_{L,j}\right) \tag{7}$$

em relação ao conjunto de parâmetros θ onde $\mathcal{L} : \mathcal{R}^{N_L} \times \mathcal{R}^{N_L} \rightarrow \mathcal{R}$ é a função de perda. Em outras palavras, o processo de aprendizado de uma rede neural profunda resume-se em alcançar bons conjuntos de parâmetros θ de forma a minimizar Eq. (7), usando algum algoritmo de otimização combinado com o método de retropropagação (Goodfellow et al., 2016).

4 Decodificação usando redes neurais

O uso de redes neurais para a tarefa de decodificação em sistemas de comunicação não é algo recente. Conforme acentua Kamassury (2020), ainda na década de 90 alguns trabalhos já se dedicavam ao tema. Caid and Means (1990), por exemplo, propuseram uma arquitetura com n entradas, uma única camada oculta e k saídas, onde a NN é projetada para prever diretamente a mensagem transmitida $\mathbf{u}$ a partir do vetor recebido $\mathbf{r}$ onde nenhuma suposição sobre as estatísticas do ruído do canal é necessária, pois a NN é teoricamente capaz de aprender o mapeamento ou extrair as características do canal a partir do treinamento (Gruber et al., 2017).

Tallini and Cull (1995), por sua vez, projetaram uma NN com duas camadas ocultas que usa o vetor síndrome (em vez do vetor recebido) como entrada da rede cuja saída é a estimativa do padrão de erro (em vez da palavra código).

Com o progresso das NNs, os algoritmos de otimização, computação paralela e as GPUs (*graphical processing units*), o interesse no desenvolvimento de novos decodificadores foi retomado em meados da última década com os trabalhos de Nachmani et al. (2016), Gruber et al. (2017), Lugosch and Gross (2018), Bennatan et al. (2018), dentre outros.

A Fig. 4 ilustra um esquema genérico para decodificadores inteligentes onde constam os blocos/estágios de codificação, modulação, canal de transmissão (com adição de ruído), pré-processamento (*log-likehood ratio*, vetor síndrome, módulo, etc.), NN e pós-processamento.

4.1 Decodificação inteligente baseado no vetor síndrome

De modo geral, consideremos um sistema de comunicação cuja transmissão usa um código linear $\mathcal{C}$ com taxa de comunicação $\frac{k}{n}$, onde uma mensagem original de k bits denotada por $\mathbf{u} \in \mathrm{GF}(2)^k$ é codificada para uma palavra código c de n bits aplicando a Eq. (1).

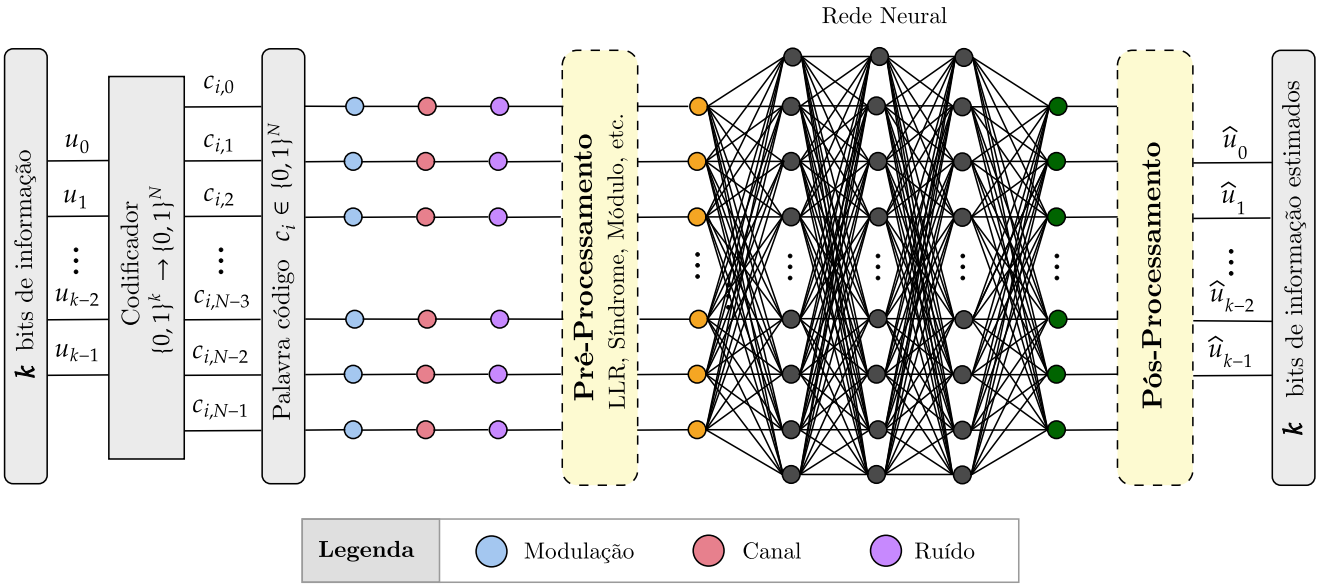

Figura 4: Esquema geral para decodificação inteligente usando redes neurais

4.2 Canal AWGN

Para uma transmissão usando o canal AWGN (*additive white gaussian noise*), o sistema de comunicação pode ser simplificadamente representado pela Fig. 5, onde é aplicado um esquema modulação BPSK (*binary phase shift keying*), e o mapeamento da palavra código para o formato bipolar é dado por:

$$\mathbf{x} = 1 - 2\mathbf{c}\,, \quad \mathbf{x} \in \{-1, +1\}^n \tag{8}$$

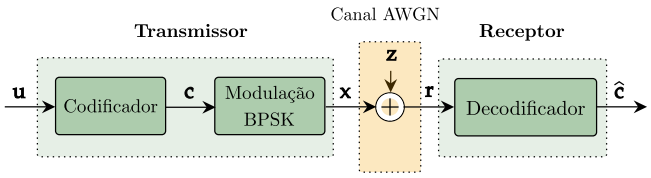

Figura 5: Sistema de comunicação usando o canal AWGN

Uma vez transmitida, o vetor recebido $\mathbf{r}$ é expresso por

$$\mathbf{r} = \mathbf{x} + \mathbf{z}\,, \quad \mathbf{z} \sim \mathcal{N}\left(0, \sigma^2\right) \tag{9}$$

onde σ^2 corresponde à variância e $\mathbf{z}$ é estatisticamente independente de $\mathbf{x}$.

Baseado nas configurações deste sistema de codificação e transmissão, Bennatan et al. (2018) projetaram uma estrutura de decodificação inteligente baseado na estimação do erro padrão a partir das informações derivadas do vetor síndrome (vide Fig. 6). Neste decodificador, $|\mathbf{r}|$ corresponde ao valor absoluto de $\mathbf{r}$ e $\mathbf{r}_b$ é obtido por:

$$\mathbf{r}_b = \frac{1 - \text{sign}(\mathbf{r})}{2} \tag{10}$$

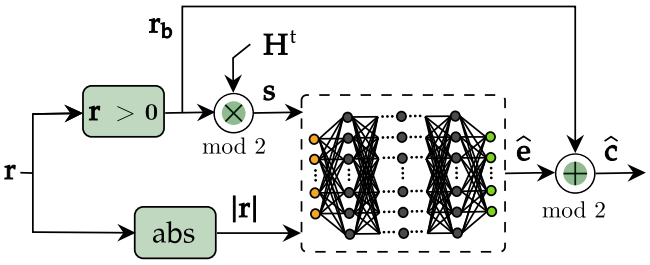

Figura 6: Estrutura de decodificação projetada por Bennatan et al. (2018)

Na prática, a estrutura deste decodificador compreende três etapas, a saber:

- Pré-processamento: estágio no qual são computados $|\mathbf{r}|$ e $\mathbf{r}_b\mathbf{H}^t$ (síndrome), onde $\mathbf{H}$ é a matriz de verificação de paridade do código $\mathcal{C}$ descrito em seções anteriores;
- Estimativa do ruido: a NN estima $\hat{\mathbf{e}}$ a partir das entradas $|\mathbf{r}|$ e $\mathbf{r}_b\mathbf{H}^t$;
- Pós-processamento: realiza-se a soma $\mathbf{r}_b + \hat{\mathbf{e}}$ em módulo 2, admitindo o resultado como a estimativa de $\hat{\mathbf{c}}$.

5 Decodificador neural para o canal AWGN

Recentemente, Kamassury (2021) apresentou uma estratégia de decodificação baseado em síndrome para um canal BSC, estendendo o trabalho de Tallini and Cull (1995). Neste trabalho, identifica-se que limitar o conjunto de treinamento da NN para exemplos com apenas padrões de erro com no máximo a capacidade de correção de erro do código, torna improvável que a NN aprenda a prever padrões de erro com erros maiores, o que justificaria seu uso para superar um decodificador *bounded-distance* (Lin and Costello Jr, 2004).

Além do mais, inconsistências dos exemplos de treinamento afetam o desempenho do decodificador.

Para uma breve compreensão dessa inconsistência, no caso do canal BSC, considere-se a seguinte palavra-código $c' \in \mathcal{C}$ com $n = 15$ bits

$$\mathbf{c}' = [0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0]$$

em dois diferentes cenários.

No cenário 1, suponha que os dois possíveis padrões de erro são

$$\mathbf{e}_1^1 = [0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0]$$
$$\mathbf{e}_2^1 = [0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0]$$

com síndromes $\mathbf{s}_1^1 = \mathbf{s}_2^1$ onde os pesos de Hamming são $w(\mathbf{e}_1^1) < w(\mathbf{e}_2^1)$. Neste caso, para duas entradas iguais ($\mathbf{s}_1^1$ e s_2^1) a NN deve predizer corretamente entre $\hat{\mathbf{e}}_1^1$ e $\hat{\mathbf{e}}_2^1$. Implicitamente, apesar do desempenho da predição ser afetado por essa inconsistência, como os exemplos de $\mathbf{e}_1^1$ ocorrem com mais frequência que $\mathbf{e}_2^1$ (por ter peso de Hamming menor), o desempenho da NN não é tão afetado visto que a rede não está sujeita muitas vezes a indecisão diante desses exemplos.

No cenário 2, suponha os seguintes padrões de erro

$$\mathbf{e}_1^1 = [0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0]$$
$$\mathbf{e}_2^1 = [0\,0\,0\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0]$$

com síndromes $\mathbf{s}_1^1 = \mathbf{s}_2^1$ onde os pesos de Hamming são $w(\mathbf{e}_1^1) = w(\mathbf{e}_2^1)$. Neste caso, como os exemplos de síndromes contribuem igualmente no treinamento (ocorrem com frequência similar), a predição neural tende a falhar consideravelmente em estimar o padrão de erro mais provável ($\hat{\mathbf{e}}_1^1$ ou $\hat{\mathbf{e}}_2^1$).

Em geral, esse comportamento deve-se ao fato de a NN está modelando a probabilidade *a posteriori bitwise*

$$\hat{\mathbf{e}} \approx \mathcal{P}\left[\mathbf{e}_j = 1 | \mathbf{s}\right] \tag{11}$$

em que, exige-se que a rede encontre todas as posições dos bits errôneos de uma única vez.

Reconhecendo esse panorama, propõe-se um algoritmo de decodificação iterativa, no qual, a NN só precisa estimar uma única posição do padrão de erro por vez. Na prática, estendemos a abordagem proposta por Kamassury (2021), intitulado decodificador *Neural-Max* (NM) para um modelo de canal de comunicação mais realista usando a estrutura apresentada na Fig. 6.

O algoritmo proposto para o canal AWGN contempla as seguintes etapas (vide Fig. 7):

1) Inicialmente, realiza-se uma decisão hard ($\mathbf{r}_b$) do vetor recebido $\mathbf{r}$ e calcula-se a síndrome $\mathbf{s}$ correspondente (Equação 3);
2) Avalia-se a condição $\mathbf{s} = 0$. Caso seja verdadeira, retorna-se $\mathbf{c} = \mathbf{r}_b$. Caso contrário, estima-se $\hat{\mathbf{e}}$ usando a NN previamente treinada;
3) Seleciona-se a posição j do maior valor predito do padrão de erro $\hat{\mathbf{e}}$;
4) Atualiza-se o vetor $\mathbf{r}$ invertendo o valor da posição j no vetor $\mathbf{r}$. Os valores das demais posições do vetor $\mathbf{r}$ são mantidos;
5) Em seguida, toma-se uma nova decisão hard sobre $\mathbf{r}$ atualizado e computa-se a síndrome;
6) Por fim, avalia-se novamente $\mathbf{s} = 0$, finalizando a iteração se a condição for atendida (admitindo $\mathbf{c} = \mathbf{r}_b$) ou repetindo T vezes o processo até um determinado critério de parada.

A complexidade desse decodificador está associada ao número de iterações T (na etapa de inferência) de forma que o número médio de iterações é superiormente limitado por

$$1 + \mathcal{P}\left[\mathcal{E}_i\right] + \cdots + \mathcal{P}\left[\mathcal{E}_1, \cdots, \mathcal{E}_{T-1}\right] \leq 1 + \sum_{i=1}^{T-1} \mathcal{P}\left[\mathcal{E}_i\right] \tag{12}$$

onde $\mathcal{P}\left[\mathcal{E}_i\right]$ é a probabilidade de erro de bloco do decodificador com i iterações. Logo, em comparação com decodificador da Fig. 6, o aumento relativo na complexidade média é muito pequeno, tornando-o interessante para implementações em hardware.

6 Experimentos e resultados

Nesta seção, investiga-se a performance em termos das métricas BER (*bit error rate*) e BLER (*block error rate*) a decodificação dos códigos BCH listados na Tabela 1 usando o esquema de decodificação iterativa proposto. Para efeitos de análise, compara-se, quando aplicável, com os melhores resultados obtidos por Lugosch and Gross (2018) e Be'ery et al. (2020), além dos resultados de Kavvousanos and Paliouras (2019) que, por sua vez, faz uso de uma das NNs menos complexas da literatura para a tarefa de decodificação.

Tabela 1: Códigos avaliados

Código (n, k)	$r = k/n$	t	nº de parâmetros treináveis
BCH(31,16)	≈ 0.52	3	67.441
BCH(31,26)	≈ 0.84	1	36.221
BCH(63,36)	≈ 0.57	5	249.705
BCH(63,39)	≈ 0.62	4	229.347
BCH(63,45)	≈ 0.71	3	191.223
BCH(63,51)	≈ 0.81	2	156.555
BCH(127,64)	≈ 0.50	10	1.129.777

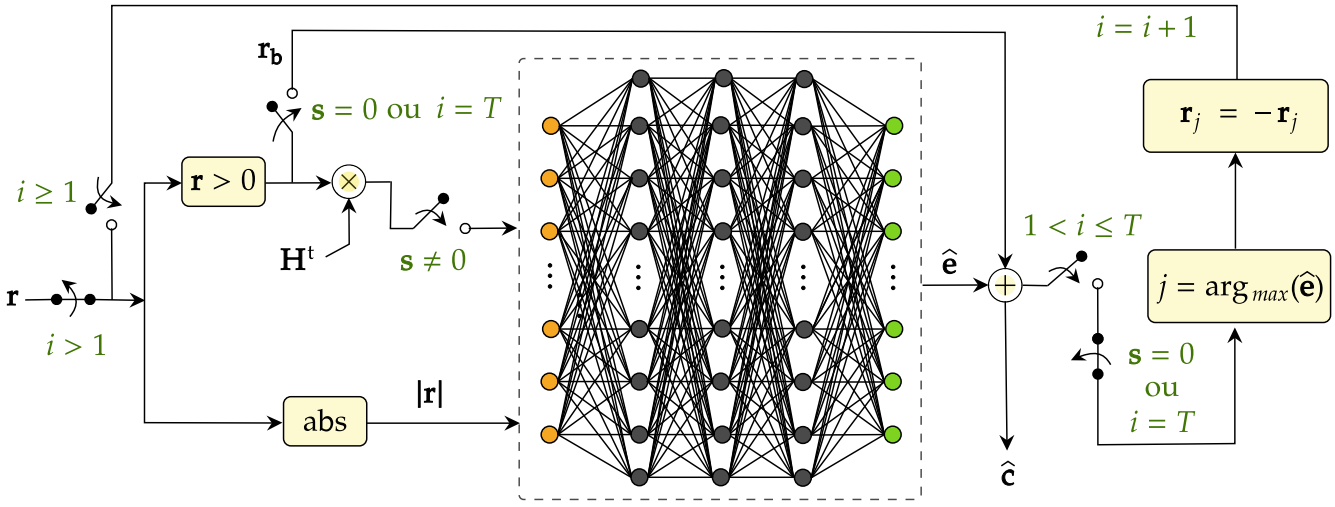

Figura 7: Estratégia proposta para decodificação iterativa de códigos transmitidos via canal AWGN.

Para a predição do padrão de erro, treinou-se a NN esquematizado na Fig. 8, constituída de 5 camadas densas ocultas com $5n - 3k$ neurônios cada, onde se aplica a tangente hiperbólica como função de ativação.

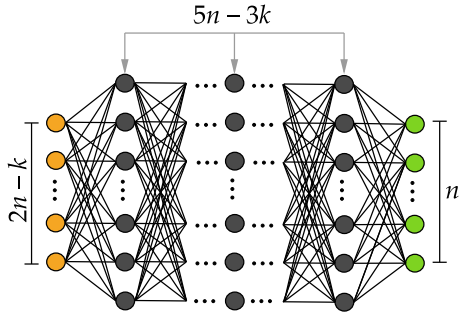

Figura 8: Esquema da NN usada nos experimentos

No caso da camada de saída (com n neurônios), utiliza-se a função sigmoide. Ademais, para acelerar e estabilizar o treinamento, todas as camadas ocultas são acompanhadas por camadas de normalização em lote (Ioffe and Szegedy, 2015).

Em termos de parâmetros treináveis, a NN da Fig. 8 possui $\approx \frac{1}{3}$ dos parâmetros da rede proposta por Kavvousanos and Paliouras (2019), o que permite usar o decodificador iterativo com um acréscimo médio de complexidade que ainda é menor que a NN original usada por Kavvousanos and Paliouras (2019).

Para o treinamento da NN, utilizou-se 10^7 exemplos (gerados em tempo real) com $\frac{E_b}{N_0} = 4$ dB que permite um bom equilíbrio entre ruído e a estrutura do código nos exemplos gerados (Gruber et al., 2017). Em relação à inicialização dos pesos da NN, emprega-se glorot normal, enquanto que para a atualização destes pesos, recorre-se ao algoritmo Adam com tamanhos de lote de 2048 e a entropia cruzada binária como função de perda (Goodfellow et al., 2016).

Seguindo o mesmo procedimento adotado por Kavvousanos and Paliouras (2019), a taxa de aprendizado é inicializada em 10^{-3}, sendo reduzida por um fator de 10^{-1} quando a perda de validação para de reduzir após 5 épocas. No estágio de inferência, estimamos a métrica BLER (e a métrica BER, numa condição subjacente) executando simulações de Monte Carlo até a ocorrência de no mínimo 100 erros de bloco para cada $\frac{E_b}{N_0}$.

Menciona-se que todas as implementações foram feitas na plataforma Colab (*google colaboratory*)[4] usando a GPU Tesla K80 e que os processos de treinamento e teste da NN foram realizados empregando a biblioteca Keras[5] com Tensorflow como *backend*.

Os desempenhos do NM para os códigos BCH que têm recebido mais atenção no contexto dos códigos curtos[6] estão dispostos na Fig. 9. Observa-se que com exceção do código BCH (127,64), os desempenhos de decodificação obtidos exclusivamente com a NN[7] superam ou tangenciam os melhores resultados dos demais trabalhos[8]. De todo modo, para todos os casos, aplicando o NM com $T \in [2, 4]$, verifica-se graduais ganhos de codificação em relação ao próprio desempenho da NN projetada.

Em relação ao código BCH(63, 36), alcança-se até 1.1 dB para BLER = 10^{-2}, enquanto que para a métrica BER em 10^{-3}, obtém-se cerca de 0.6 dB. Já no caso do código BCH(63, 45), em comparação com os melhores resultados de Kavvousanos and Paliouras (2019), há ganhos de codificação aproximados de 0.6 dB e 0.3 dB para BLER = 10^{-2} e BER = 10^{-3}, respectivamente.

[4] https://colab.research.google.com/
[5] https://keras.io/
[6] Nesta figura, para os códigos BCH com $n = 63$, também estão inclusas as curvas teóricas HDD-HD e ML disponibilizadas por Helmling et al. (2019).
[7] Equivalente a $T = 1$.
[8] Para os casos que possuem resultados disponíveis na literatura.

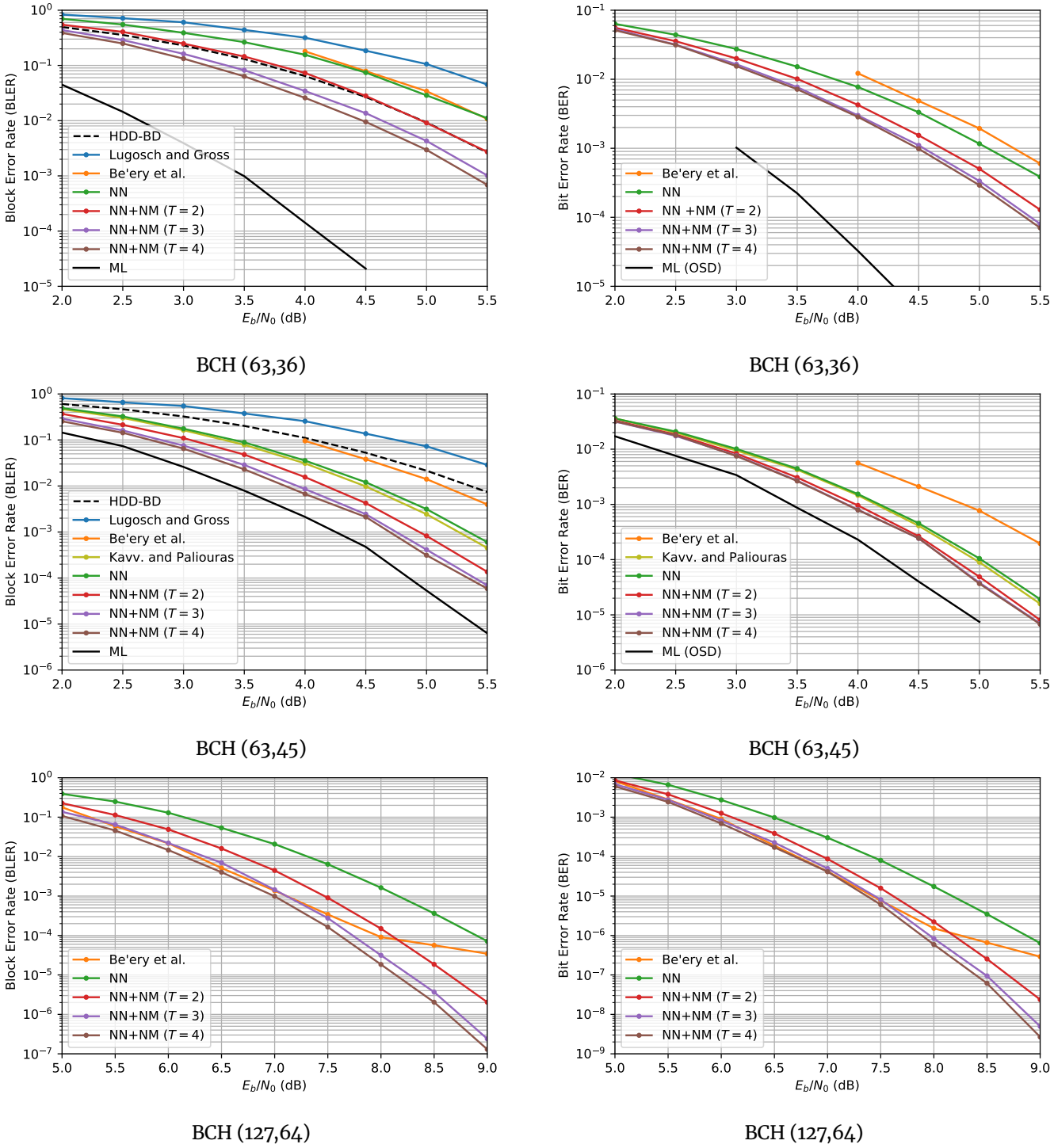

Figura 9: Desempenhos obtidos com o decodificador de Bennatan et al. (2018) usando a NN da figura Fig. 8 e o NM para os códigos BCH mais investigados na literatura.

Especificamente para o código BCH(127,64), visualiza-se que quando $\frac{E_b}{N_0} < 7$ dB, o NM (com $T = 4$) alcança resultados muito próximos daqueles obtidos por Be'ery et al. (2020). Além disso, à medida que $\frac{E_b}{N_0}$ ultrapassa 7 dB, cada vez menos iterações são necessárias para superar os resultados de Be'ery et al. (2020).

Por fim, objetivando avaliar ainda mais as performances do NM para códigos BCH com comprimentos diferentes, estendeu-se o uso deste decodificador para os demais

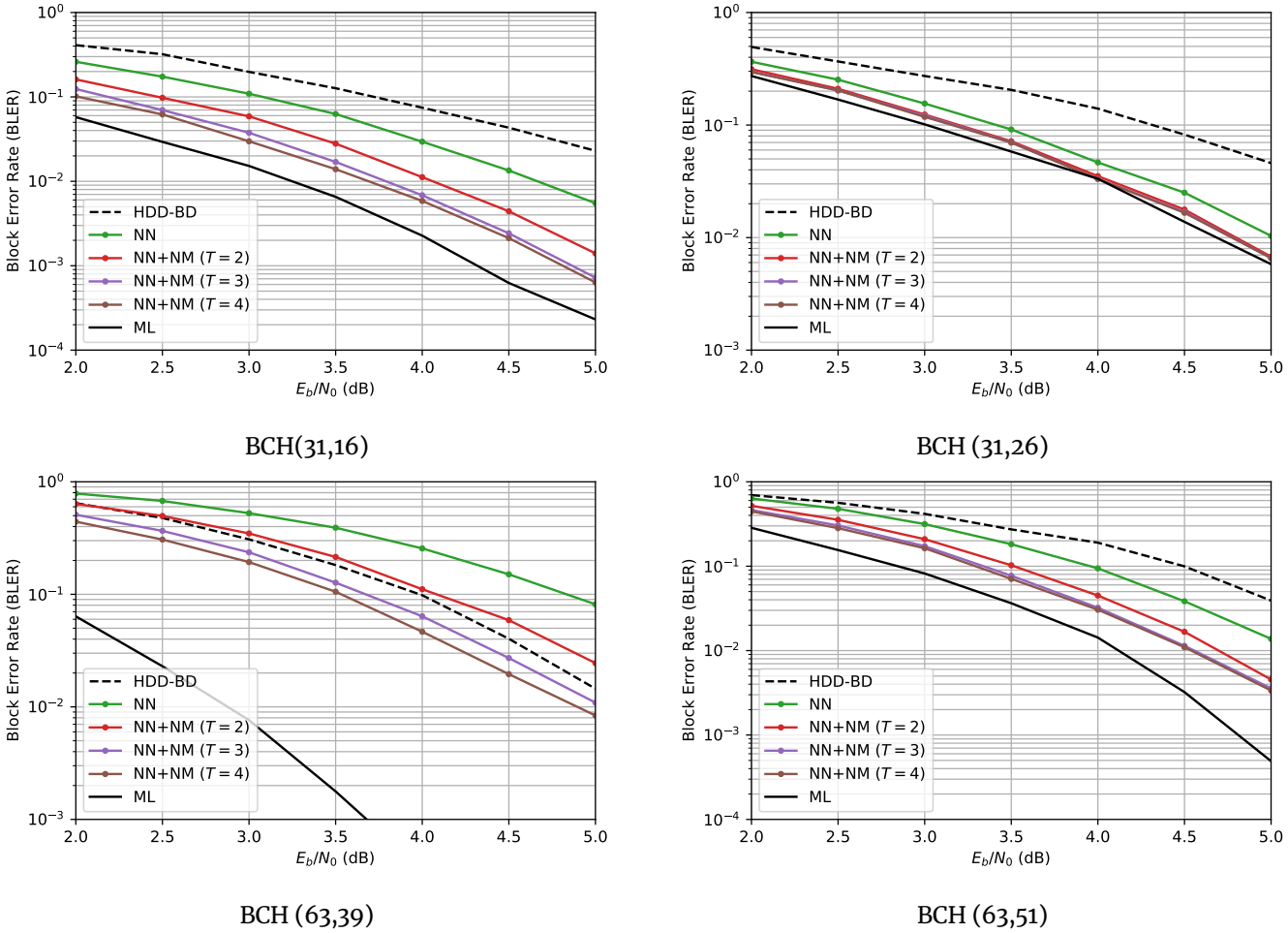

Figura 10: Desempenhos obtidos com o decodificador de Bennatan et al. (2018) e o NM para os demais códigos BCH da Tabela 1 aplicando a rede neural da Fig. 8.

códigos listados na Tabela 1. A Fig. 10 ilustra os desempenhos[9] do NM para esses códigos, na qual, novamente atestamos que para todos, o decodificador iterativo supera os resultados obtidos exclusivamente com a rede neural.

Experimentalmente, observou-se que para todos os códigos, com $T \geq 4$ não se observa melhorias significativas usando o decodificador proposto.

7 Considerações finais

No presente artigo, apresentou-se um novo esquema de decodificação que seleciona iterativamente as posições mais confiáveis para serem os bits errôneos do padrão de erro, atualizando o vetor recebido (no lado do receptor) quando novas posições são selecionadas. Além disso, projetou-se uma nova rede neural densa com reduzida complexidade para atuar em sistemas de decodificação que usam as informações de confiabilidade e do vetor síndrome como entrada.

Os resultados obtidos evidenciam que ao custo de um moderado incremento de complexidade, o decodificador iterativo supera os desempenhos disponíveis na literatura com a vantagem de ser flexível para ser empregado em qualquer estrutura de decodificação baseada no vetor síndrome.

[9] A análise dos ganhos de codificação obtidos em relação a BLER são mais interessantes do que a BER especialmente para aplicações mais recentes. Por exemplo, para comunicação baseada em IP, os pacotes com violações de CRC são descartados, exigindo que a camada física do sistema de comunicação entregue pacotes sem erros de bit. Isso também é válido para o contexto das redes 4G/5G cujos requisitos de confiabilidade são especificados em termos dessa métrica.

Referências

Bennatan, A., Choukroun, Y. and Kisilev, P. (2018). Deep learning for decoding of linear codes - a syndrome-based approach, *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 1595–1599. http://dx.doi.org/10.1109/ISIT.2018.8437530.

Bennis, M., Debbah, M. and Poor, H. V. (2018). Ultrareliable and Low-Latency Wireless Communication: Tail, Risk,

and Scale, *Proceedings of the IEEE* **106**(10): 1834–1853. http://dx.doi.org/10.1109/JPROC.2018.2867029.

Be'ery, I., Raviv, N., Raviv, T. and Be'Ery, Y. (2020). Active deep decoding of linear codes, *IEEE Transactions on Communications* **68**(2): 728–736. http://dx.doi.org/10.1109/TCOMM.2019.2955724.

Bose, R. and Ray-Chaudhuri, D. (1960a). Further results on error correcting binary group codes, *Information and Control* **3**(3): 279–290. https://doi.org/10.1016/S0019-9958(60)90870-6.

Bose, R. and Ray-Chaudhuri, D. (1960b). On a class of error correcting binary group codes, *Information and Control* **3**(1): 68–79. https://doi.org/10.1016/S0019-9958(60)90287-4.

Caid, W. and Means, R. (1990). Neural network error correcting decoders for block and convolutional codes, *[Proceedings] GLOBECOM '90: IEEE Global Telecommunications Conference and Exhibition*, pp. 1028–1031 vol.2. http://dx.doi.org/10.1109/GLOCOM.1990.116658.

Cavalcanti, F. R. P., Maciel, T. F., Freitas Júnior, W. C. F. and Silva, Y. C. B. (2018). *Comunicação Móvel Celular*, 1 edn, Elsevier Editora, Rio de Janeiro.

Goodfellow, I., Bengio, Y. and Courville, A. (2016). *Deep Learning*, MIT Press.

Gruber, T., Cammerer, S., Hoydis, J. and t. Brink, S. (2017). On deep learning-based channel decoding, *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, pp. 1–6. http://dx.doi.org/10.1109/CISS.2017.7926071.

Haykin, S. (2014). *Digital Communications Systems*, 1 edn, John Wiley & Son, Inc., Hoboken, NJ.

Helmling, M., Scholl, S., Gensheimer, F., Dietz, T., Kraft, K., Ruzika, S. and Wehn, N. (2019). Database of Channel Codes and ML Simulation Results. www.uni-kl.de/channel-codes.

Hocquenghem, A. (1959). Codes correcteurs d'erreurs, *Chiffers* **2**: 147–156. http://kom.aau.dk/~heb/kurser/NOTER/KOFA02.PDF.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift, *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, JMLR.org, p. 448–456. https://doi.org/10.5555/3045118.3045167.

Kamassury, J. K. S. (2020). *Decodificação de códigos de comprimento curto usando redes neurais profundas*, Master in Electrical Engineering, Graduate program in electrical engineering at Federal University of Santa Catarina. Available at https://repositorio.ufsc.br/handle/123456789/216351.

Kamassury, J. K. S. (2021). Decodificador baseado em Rede Neural Profunda para Códigos de Bloco Lineares Curtos Transmitidos via Canal Binário Simétrico, *REMAT: Revista Eletrônica da Matemática* **7**(1): e3006–e3006. http://dx.doi.org/10.35819/remat2021v7i1id4389.

Kamassury, J. K. S., Tôrres, I. F. d. M. and Duarte, W. G. (2019). Decodificação de máxima verossimilhança para códigos de bloco lineares: probabilidades de erro do código de repetição e do código de Hamming, *REMAT: Revista Eletrônica da Matemática* **5**(2): 177–191. http://dx.doi.org/10.35819/remat2019v5i2id3371.

Kavvousanos, E. and Paliouras, V. (2019). Hardware implementation aspects of a syndrome-based neural network decoder for bch codes, *2019 IEEE Nordic Circuits and Systems Conference (NORCAS): NORCHIP and International Symposium of System-on-Chip (SoC)*, pp. 1–6. http://dx.doi.org/10.1109/NORCHIP.2019.8906946.

Lin, S. and Costello Jr, D. J. (2004). *Error control coding*, 2 edn, Prentice-Hall, Upper Saddle River, N.J.

Lugosch, L. and Gross, W. J. (2018). Learning from the syndrome, *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, pp. 594–598. http://dx.doi.org/10.1109/ACSSC.2018.8645388.

Nachmani, E., Be'ery, Y. and Burshtein, D. (2016). Learning to decode linear codes using deep learning, *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pp. 341–346. http://dx.doi.org/10.1109/ALLERTON.2016.7852251.

O'Shea, T. and Hoydis, J. (2017). An introduction to deep learning for the physical layer, *IEEE Transactions on Cognitive Communications and Networking* **3**(4): 563–575. http://dx.doi.org/10.1109/TCCN.2017.2758370.

Shirvanimoghaddam, M., Mohammadi, M. S., Abbas, R., Minja, A., Yue, C., Matuz, B., Han, G., Lin, Z., Liu, W., Li, Y., Johnson, S. and Vucetic, B. (2019). Short Block-Length Codes for Ultra-Reliable Low Latency Communications, *IEEE Communications Magazine* **57**(2): 130–137. http://dx.doi.org/10.1109/MCOM.2018.1800181.

Tallini, L. G. and Cull, P. (1995). Neural nets for decoding error-correcting codes, *IEEE Technical Applications Conference and Workshops. Northcon/95. Conference Record*, pp. 89–. http://dx.doi.org/10.1109/NORTHC.1995.485019.