Journal of Theoretical and Applied Information Technology

15th September 2023. Vol.101. No 17

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

COOPERATION OF TWO LOGISTIC REGRESSION MODELS FOR DECODING LINEAR BLOCK CODES

CHEMS EDDINE IDRISSI IMRANE[1]*, SAID NOUH[1], BELLFKIH EL MEHDI[2], MOHAMMED EL ASSAD[1] AND ABDELAZIZ MARZAK[1]

[1] LTIM Lab, Faculty of Sciences Ben M'sik, Hassan II University, Casablanca, Morocco
[2] LAMS Lab, Faculty of Sciences Ben M'sik, Hassan II University, Casablanca, Morocco
E-mail: imran.chems@gmail.com

ABSTRACT

Error-correcting codes (ECCs) play a vital role in protecting data against corruption in storage systems and random errors due to noise effects in communication channels. Communication protocols typically employ various techniques to detect and correct errors in transmitted messages. Syndrome-based decoding is one of the most widely used techniques for decoding algorithms. In recent years, machine learning has gained attention for building efficient decoders that demonstrate promising results, offering new avenues for error correction. In this paper, we propose improving our logistic regression-based decoder utilizing two distinct models (2LRDec). The key distinction between LRDEC and 2LRDEC is how they manage decoding tasks. While both use logistic regression models for decoding, 2LRDEC specifically employs two distinct models, which significantly enhance its performance, especially for codes with larger lengths (n). Another critical advantage of 2LRDEC is its reduced training and execution complexity. Splitting the data and using two models in parallel, allows for faster training and execution, thus increasing overall decoding efficiency. This makes 2LRDEC an effective and efficient solution for handling larger code lengths and real-time decoding scenarios.

Keywords: *Error Correcting Code; Syndrome Decoding; Machine Learning; Logistic Regression*.

1. INTRODUCTION

There are multiple perspectives to deal with enhanced communication reliability. Recently, many papers have treated telecommunication technologies as the main subject, aiming to improve their efficiency and speed and guarantee reliable communication between all connected objects. For this reason, and since no channel is noiseless, researchers get involved against the effect of signal noise. Some conducted studies that use deep learning (DL) in decoding algorithms to correct any possible errors, reducing complexity and increasing efficiency, in Figure 1 we show how the channel noise affected the transferred message.

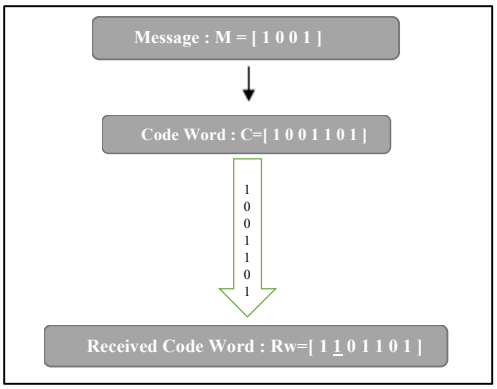

Figure 1: Example Of Transmitted Message In A Noisy Channel

The redundant-based coding technique (error-correcting codes EECs) is intended to correct errors in the transmission of information on an unreliable communication channel which is classified into two

major types: The block codes, which encode data using block independently, and the convolutional codes; that is processed not just in connection to the channel encoder's recent input, but also with the last entry blocks. There are two main classes of error-correcting codes: Convolutional codes and block codes. Convolutional codes are generated by passing the input data through a shift register, where the output depends on the current input as well as the previous inputs. On the other hand, block codes divide the input data into fixed-size blocks and encode each block independently using a predefined linear transformation.

While both convolutional and block codes shown in Figure 2 have their unique advantages and applications, our study will focus on linear block codes. Linear block codes, a subset of block codes, have a rich algebraic structure that enables efficient encoding and decoding algorithms. This makes them an attractive choice for various communication systems, as they offer a good balance between error-correcting capabilities and computational complexity. Examples of linear block codes include Reed-Solomon codes, Hamming codes, and Bose-Chaudhuri-Hocquenghem (BCH) codes.

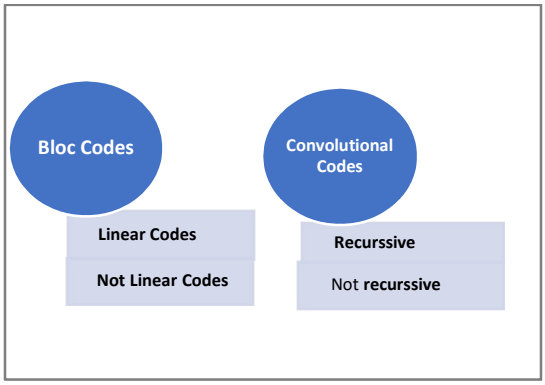

Figure 2: ECC Types

We are focusing again on the decoding part, more precisely, on syndromes calculation where we use the machine learning (ML) technique in proposing our decoder the LR-based method.

In Table 1, we provide a concise overview of the different decoder types based on the SIHO, HIHO, HISO, and SISO classifications, along with examples and applications for each decoder type. Note that some decoders may not fit neatly into one of these categories and that there might be variations or hybrid approaches depending on the specific application and system requirements. In this work, we also explore the applicability and performance of our proposed 2LRDec decoder in the context of hard-input hard-output (HIHO) channels. In this respect, HIHO channels are a class of communication channels where both the input and output signals are quantized into discrete levels, making them particularly relevant for digital communication systems. The error-correction process in HIHO channels typically involves making hard decisions about the transmitted symbols based on the received noisy signals.

Table 1: Input-Output type for decoding

Decoder Type	Input	Output	Examples and Applications
SIHO	Soft Input	Hard Output	Iterative decoders for Turbo codes and LDPC codes; decoders that use soft information for improved performance
HIHO	Hard input	Hard Output	Iterative decoders for Turbo codes and LDPC codes; decoders that use soft information for improved performance
HISO	Hard Input	Soft Output	Decoders are used as intermediate stages in iterative decoding processes or concatenated coding schemes
SISO	Soft Input	Soft Output	Iterative decoders for Turbo codes and LDPC codes; decoders that exchange soft information between stages

By extending our logistic regression-based decoding approach to the HIHO channel setting, we aim to assess the decoder's adaptability to different types of communication environments and evaluate its potential for broader applicability. To make the work much more scrutinized, we will conduct a series of experiments and analyses to investigate the performance of the 2LRDec decoder under HIHO channel conditions, comparing it with other state-of-the-art decoders designed for these channels.

Numerous studies have explored the integration of deep learning algorithms with error correction codes (ECC). These studies take advantage of the attributes of linear codes, combining them with the functionality of deep learning algorithms to create

decoders for BCH and QR codes. A decoder designed for polar codes, leveraging deep neural networks (DNN), has also shown exceptional efficiency. To improve the belief propagation algorithm, BCH codes are being treated with a deep learning approach. Researchers have harnessed machine learning algorithms in synergy with syndrome calculation to improve decoder performance, not only in terms of bit error rate (BER) but also in time complexity. This method is particularly applicable to linear block codes. In our last work concerning the single logistic regression model, we are confronted with big difficulties in completing the training phase to build the decoding model LRDec [1] for the BCH codes with n >=63. For this reason, the idea is to create two models by splitting the dataset into training both codes' syndromes and the errors to minimize the size of the data for each model and reduce the complexity of training.

This research paper is organized into several coherent sections that follow the introduction. The second section dives into the established work in the field of decoding techniques, providing a foundation and context for our research. Subsequently, in section 3, we explore the burgeoning field of machine learning techniques and their application to our problem area. Our primary contribution, the novel 2LRDec decoder, is presented in Section 4. This section details the architecture of the 2LRDec decoder and its model creation and decoding algorithm, thoroughly explaining the thought process and technicalities involved in its development. The final section brings to light the results and solutions of our experiments and comparisons of performance between our 2LRDec decoder and other existing decoders. This comparison not only considers their respective performance but also juxtaposes the complexity of our proposed decoder against others.

2. RELATED WORKS

We are truly convinced that the errors problem in the error-correcting codes (ECCs) field is always present; that is why conceiving, enhancing, and ameliorating decoders in terms of complexity or bit error rate (BER), or both is a must. Different approaches, techniques, and methods exist based on different aspects, but we can classify them into three main classes: Algebraic decoders, meta-heuristic decoders, and machine learning-based decoders. Our research paper is framed in the class of machine learning-based decoders.

Many kinds of research have dealt with machine learning techniques, in which a breakthrough has been presented on the level of complexity, BER. We cite here, Chemseddine Idirissi et al. [1], a machine learning technique–based decoder, uses a multiclass regression model for finding errors from syndromes in linear codes like some BCH codes, and some QR codes. Another work made by Nouh et al. [2] used syndrome decoding using a genetic algorithm. In the research paper of Moulay Seddiq et al. [3], they used a hash table to optimize syndrome decoding. The work of Nachmani et al. [4], where the belief propagation method is improved using a deep learning method. Furthermore, by adding weights to the edges of the Tanner graph, this technique generalizes the standard belief propagation algorithm. Also, in another research [5], it is demonstrated that, despite the vast example space, deep learning approaches may be employed to improve a typical belief propagation decoder. The min-sum method has shown similar advances. It is also demonstrated that linking the decoder parameters across iterations to construct a recurrent neural network design may be done with equivalent results. It is shown in their new decoder, which decodes linear block codes using a recurrent neural network design. When compared to a feed-forward neural network with significantly fewer parameters, it has exhibited equivalent BER outcomes [6]. Again, Nachmani et al. [7], find that a merging allows the decoder to attain near maximum likelihood performance for High-Density Parity Check codes using a novel decoder based on the neural Belief Propagation algorithm and the Automorphism Group. Lugosch and Gross [8] presented in their paper an offset min-sum decoding augmented with learnable offset parameters. This technique employs no multiplications and has a parameter count less than half that of the multiplicative algorithm, thus, making it a more hardware-friendly approach. Later, in work made by Lugosch and Gross [9], The syndrome loss, based on a relaxation of the syndrome, is presented as an alternative loss function for neural error-correcting decoders. The syndrome loss penalizes the decoder if it produces outputs that do not match valid codewords. Training with the syndrome loss has been proven to result in decoders with consistently decreased frame error rates for various short block

codes, with minimal cost during training and no higher cost during inference.

Unlike other approaches, a unique framework for using deep neural networks (DNN) for soft decoding of linear codes at unlimited block lengths, allows for unrestricted DNN creation and the use of solid designs discovered in other contexts [10]. Moreover, Capri et al. [11] have reviewed many iterative decoding techniques that require a decision-making process at each step, including bit flipping (BF) decoding, residual belief propagation, anchor decoding, and applied reinforcement learning to find successful decoding strategies for binary linear codes. They then show how to translate such algorithms to Markov decision processes. Instead of relying on heuristics or intuition, data-driven learning of optimum choice techniques is possible, In [12] authors review the application of deep learning neural networks for decoding high-density parity check (HDPC) codes, such as BCH and RS codes, in satellite communications. A decoder system model for satellite communication is presented, which uses three neural networks to compensate for non-white noise, shortest circles in the Tanner graph, and unreliable information. They propose various neural decoders with permutation invariant structures for BCH and punctured RM codes. The cyclically equivariant and affine equivariant neural decoders are introduced, both outperforming previous neural decoders for cyclic codes. The affine decoder has a lower error probability but a longer runtime. Additionally, a list decoding version of the cyclic decoder significantly reduces the frame error rate for these codes, with some high-rate codes having less than a 0.1 dB gap to the Maximum Likelihood decoder[13].

The other decoding methods, especially in the heuristic and met-heuristic methods have recognized the newest research papers. In the field of polar codes, the authors in [14] utilize a permutation matrix between the code words. The proposed adaptation achieves maximum likelihood decoding performance while maintaining the low decoding complexity of polar codes, outperforming other universal decoders of linear codes. A new efficient hard decoding algorithm for binary systematic Quadratic Residue (QR) codes that correct $erroneous$ bits or less in the received word. The proposed decoder significantly decreases decoding complexity without performance loss, making it a strong competitor for decoding QR codes with various lengths [15]. [16] This research investigates two decoders with reduced complexity and strong error correction performance in AWGN channels. Applied to linear block codes like BCH and QR codes, the decoders efficiently improve performance in terms of BER. For instance, decoding BCH (31, 11, 11) using SDHT results in a coding gain of 34.5 dB and a 75% error rate reduction compared to uncoded transmissions.

The problem of decoding linear block codes using machine learning techniques involves developing efficient and accurate methods to decode received codewords that have been corrupted by channel noise. Traditional decoding algorithms for linear block codes, are often computationally intensive and may not provide optimal performance in practical scenarios. Machine learning-based decoding aims to address this issue by leveraging the power of artificial intelligence to enhance decoding accuracy while potentially reducing computational complexity.

"What novel feature engineering, regularization, and optimization strategies can be integrated into Logistic Regression Decoder [1] to enhance its decoding capabilities, achieve higher predictive performance, and manage model complexity in complex datasets?". This question focuses on exploring various methods to improve the performance and complexity of logistic regression for decoding purposes while considering the impact of feature engineering, and optimization techniques. It highlights the need to address challenges posed by intricate datasets and aims to advance the understanding of logistic regression's potential in decoding applications.

3. MACHINE LEARNING TECHNIQUES:

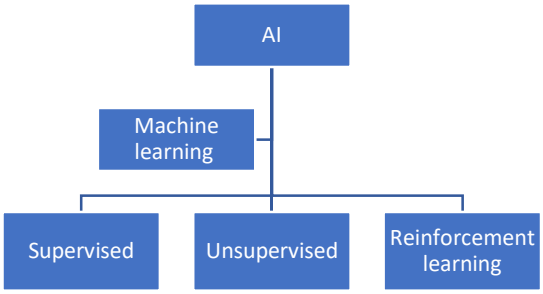

Figure 3: Machine Learning Techniques

Machine learning, a subset of artificial intelligence, encapsulates the capability of computer systems to independently discover solutions by recognizing patterns within vast databases. In essence, it serves as an umbrella term for a collection of methodologies and tools that empower computers to learn and adapt autonomously.

The machine learning algorithms are classified into three categories, depending on the data specifications, supervised learning which is applied when each training data items have labels, then unsupervised learning is when we focus on grouping data into similar groups or categories, and finally, reinforcement learning, which presents a big difference to the other two categories. This is because we can build machine learning models out of training and testing data. It works by exploring the system environment and making decisions by an agent.

In this paper, we are going to use supervised machine learning algorithms, mainly the logistic regression model. This kind of model predicts whether an event will occur (value of 1) or not (value of 0) from the optimization of the regression coefficients. This result always varies between 0 and 1. When the predicted value is greater than 0.5, the event is likely to occur, whereas when this value is less than 0.5, it is not.

4. THE PROPOSED DECODER 2LRDEC:

4.1. Prepairing Data

After beginning to create our decoding model, we must prepare the database.

$$f(X) = Y$$

In the following of this paper,

- X will present the set of the syndrome errors in the binary form for the code $C(n,k,t)$.

$$\boldsymbol{X} = \{\boldsymbol{x_i} / \boldsymbol{x_i} = [\boldsymbol{s_1}, \boldsymbol{s_2}, .. \boldsymbol{s_{n-k}}]\} \quad \boldsymbol{s_j} \in \{\mathbf{0}; \mathbf{1}\}$$

- Y will present the set of all possible errors y_i for the code $C(n,k,t)$.

$$\boldsymbol{Y} = \{\boldsymbol{y_i} / \boldsymbol{y_i} = [\boldsymbol{b_1}, \boldsymbol{b_2}, ... \boldsymbol{b_n}]\} \quad \boldsymbol{b_j} \in \{\mathbf{0}; \mathbf{1}\}$$

4.2. Phase 1: Creating the model.

We present our new model called 2LRDec which consists of three major stages in the first phase (creation of the model) as illustrated in Fig. 1.

- Stage 1: Generating all data of the linear code syndromes: X and their related errors: Y.
- Stage 2: Split the data base into $(X_1|Y_1)$, and $(X_2|Y_2)$.
- Stage 3: Training the models LRM_1 using $(X_1|Y_1)$, and LRM_2 using $(X_2|Y_2)$.

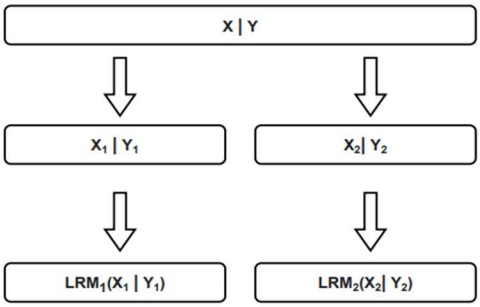

Figure 1: 2LRDec decoder model

In this novel approach for decoding, we propose utilizing logistic regression models to minimize training complexity. The method involves splitting the data into two separate databases and training two distinct logistic regression models on each dataset. In reality, this division of data allows for parallelized training, effectively reducing the computational burden and training time. By leveraging logistic regression models, which are inherently efficient, this approach aims to optimize decoding performance while minimizing training complexity.

4.3. Phase 2: Decoding Architecture

After training the two logistic regression models, $LMR1$ and $LMR2$, the decoding process involves using both models to predict two probable errors, e_1, and e_2. To determine the correct error, we calculate the syndromes for bindicesndiIndiceese$_1$ anIndicIndicee_2. The error with the null syndrome is considered the correct error, as it indicates that the error has been effectively corrected. By employing this method, we can effectively utilize both $LMR1$ and $LMR2$ models in the decoding process, ultimately enhancing the overall decoding performance. (Fig.2).

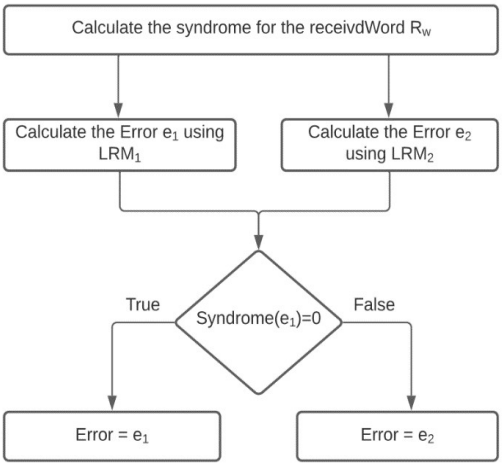

Figure 2: 2LRDec Diagram

As Follow, an algorithm for $2LRDec$ to find the correct error between e_1 and e_2 to achieve the decoding process of the $2LRDec$ decoder.

Algorithm 2 The algorithm of decoding By the 2LRDec Decoder

Require: LRM_1 Decoder and LRM_2 Decoder Models
Require: $syn()$ error syndrome calculation
$R_w \leftarrow ReceivedMessage$
$e_1 \leftarrow LRM1(R_w)$
$e_2 \leftarrow LRM2(R_w)$
$Err \leftarrow 0$
if $syn(e_1) = 0$ **then**
$Err \leftarrow e_1$
end if
if $syn(e_2) = 0$ **then**
$Err \leftarrow e_2$
end if

Figure 2: 2LRDec Algorithm

5. RESULTS AND DISCUSSION

5.1. Rresults in terms of BER

After creating the two logistic regression decoding models, 2LRDec, we apply them to an additive white Gaussian noise (AWGN) channel for decoding BCH codes with code lengths n = 15 and 31. These BCH codes have different values for k (information bits) and varying minimal distances, affecting the error-correcting capabilities (t). By using the 2LRDec models in conjunction with the AWGN channel and various BCH codes, we can analyze and evaluate their performance and effectiveness in decoding under different conditions. This approach allows for a comprehensive understanding of the models' adaptability and robustness in real-world communication scenarios.

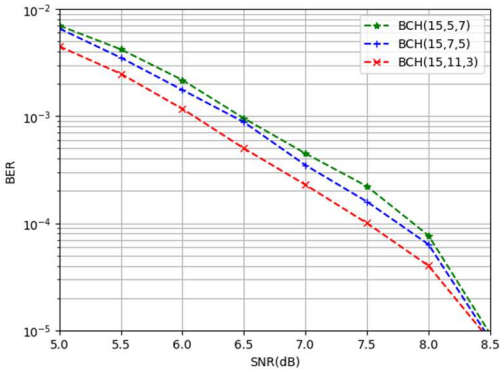

Figure 3: 2LRDEC results for n=15

In Figure 3 we present the results of the 2LRDEC decoder for BCH(15,5,7), BCH(15,7,5), and BCH(15,11,3) codes in AWGN channel transmission. We have a value of BER=10^{-5} for SNR=9.6 dB without coding-decoding techniques, in consequence, we observe that the 2LRDEC decoder for the 3 codes has approximately the same gain of 1.2 dB. All the decoding has a value of BER=10^{-5} in SNR < 8.4 dB.

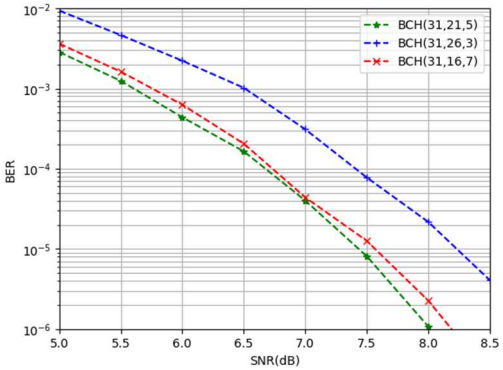

Figure 4: 2LRDEC results for n=31

The comparison of the decoding performance, using the 2LRDec approach for BCH codes of length 31, indicates a significant improvement in the Bit Error Rate (BER) when compared to the baseline scenario without a decoder. In the baseline scenario, an SNR of 9.6 dB yields a BER of 10^{-5}. However, the application of 2LRDec results in a marked

reduction of BER at a lower SNR. Specifically, at an SNR of 7.5 dB, the BCH(31,26) code achieves a BER of approximately 7.77e-05. Similarly, the BCH(31,21) and BCH(31,16) codes register BERs of about 8.11e-06 and 1.26e-05 respectively. These BERs are substantially lower than the benchmark 10^{-5} achieved without a decoder at 9.6 dB. Further improvement is noted at an SNR of 8.0 dB, where the BERs for all three codes drop to an order of magnitude lower than the 10^{-5} baseline. These results demonstrate the superior performance of the 2LRDec approach.

Those results have shown that the 2LRDec decoding approach, when applied to BCH codes, not only improves the error correction capabilities, but also achieves these improvements at a lower SNR compared to systems without decoders. This indicates the efficacy of the 2LRDec method and its potential to enhance the reliability and efficiency of data transmissions for BCH codes.

5.2. decoders Comparisons

After testing the decoder 2LRdec for different BCH codes, we have compared its results with those obtained by the decoder LRDec[1], HSDEC[3], and Nachmani[6].

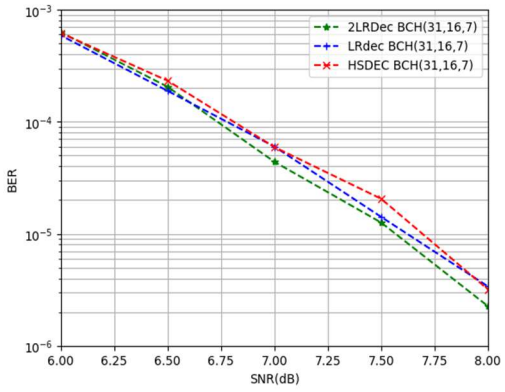

Figure 5: Performance comparison of 2RLDec, LRdec, and HSDEC for the BCH code (31,16,7).

Upon examining the performance of the 2LRDec, LRDec, and HSDec decoders on BCH codes of length 31 with a minimum distance of 7, we observe that in the lower SNR range (5 to 6.5), the 2LRDec decoder appears to provide the best performance with the lowest Bit Error Rate (BER). This demonstrates its robustness and efficacy in conditions with higher noise levels. Also, if we move to higher SNR levels (7 to 8.5), the performance differences between the three decoders become less pronounced. However, the 2LRDec decoder continues to maintain a slight advantage, achieving a lower BER compared to the LRDec and HSDec decoders.

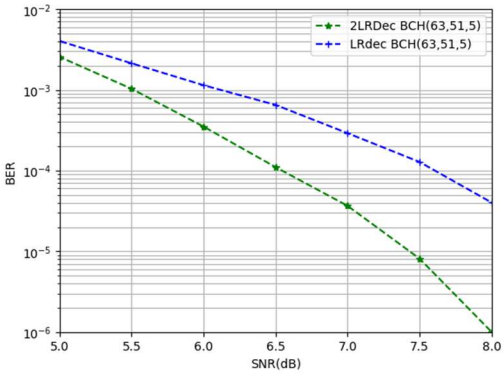

Figure 6: Performance comparison of 2RLDec and LRdec for the BCH code (63,51,5).

Upon evaluating the performance of the 2LRDec and LRDec decoders, it is evident that the 2LRDec approach demonstrates superior efficiency, particularly in low SNR scenarios. Both decoders were assessed using BCH codes with a length of 63 and a minimum distance of 5 (Figure 6). When considering low SNR values (ranging from 0.5 to 4.5), the 2LRDec method consistently achieves lower BERs compared to the LRDec approach. This suggests that the 2LRDec method is more robust and effective in noisy conditions, enhancing error correction performance in these circumstances. In conclusion, the 2LRDec approach proves to be a more efficient and robust decoding method when compared to the conventional LRDec approach. It provides superior performance, particularly at lower SNR levels, which is crucial for ensuring reliable data transmission in real-world communication scenarios often characterized by high levels of noise.

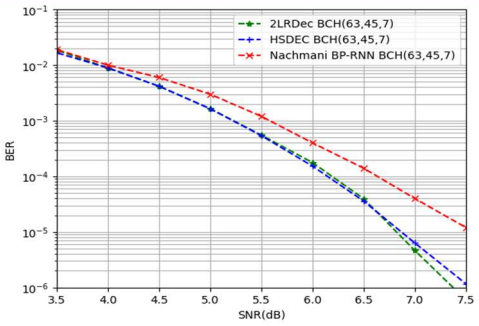

Figure 7: Performance comparison of 2RLDec, HSDEC, and NAchmani BP-RNN for the BCH code (63,45,7).

Looking at the performance of the 2LRDec, HSDec, and Nachmani decoders for BCH codes of length 63 with a minimum distance of 7, it is evident that each decoder has its strengths and weaknesses across the different Signal-to-Noise Ratio (SNR) values, at lower SNR values (2 to 4), the 2LRDec decoder exhibits superior performance, achieving the lowest Bit Error Rate (BER). This shows its robustness in noisy environments and a clear edge in error correction. As we transition to mid-range SNR values (4.5 to 6), the performance differences between the three decoders start to diminish, but the 2LRDec decoder still maintains a slight advantage in terms of lower BER. Moving towards higher SNR values (6.5 to 7.5), the 2LRDec and HSDec decoders are on par, demonstrating similar BERs, whereas the Nachmani decoder yields slightly higher BERs.

However, while assessing these decoders, it is also important to consider their complexity. The HSDec, although delivering commendable performance, has a significant space complexity due to its need to store all syndromes of codes and errors. This can be a drawback in systems with limited computational resources or memory. In contrast, the 2LRDec decoder provides comparable or even better performance and has a lower space complexity, only requiring the storage of the trained model. This makes it a more efficient and practical choice in terms of balance between performance and computational resource requirements.

5.3. Complexity study:

5.3.1 Training Complexity:

In this work, the 2LRDec decoder for the linear codes BCH presents efficient results. Table 1 shows the difference between this decoder and the LRDec in how to build o decoder-based logistic regression model with perfect accuracy in the training phase. Also, it indicates the score of some codes in the training model process.

Table 1: Model Score of 2LRDec and LRdec

Linear codes	Score Model	
	LRDec	2LRDec
BCH(15,5,7)	100%	100%
BCH(31,16,7)	100%	100%
BCH(63,51,5)	96%	100%
BCH(63,45,7)	-	100%

In this table above, we can see that the LRDec and 2LRDec decoders have an ideal training score for the BCH (15,5,7) and BCH (31,16,7) codes, but concerning the BCH (63,51,5) code, the LRDec decoder has a score of 96% lower. Good than the 2LRDec decoder, which has a score of 99%.

Generally, for the BCH codes $C(n,k,t)$ with n and n-k higher, the LRDEC decoder has difficulties creating a model with the perfect score, this is why we started by dividing the dataset (syndromes and errors) on two models (2LRDec), which shows its effectiveness on BCH with $n = 15, n = 31$ and $n = 63$ codes.

Training a multiclass logistic regression model involves learning a set of weight coefficients for each class. The number of coefficients per class is equal to the number of features ($n - k$ in our case, the length of the syndrome), plus one for the bias term. So, if we have NC classes, the total number of coefficients Nb_{coef} is

$$Nb_{coef} = \mathrm{NC}\,.\,(\mathrm{n} - \mathrm{k} + 1).$$

The training complexity for logistic regression [17] is typically

$$C_{Training} = O(\mathrm{I}\,.\,Nb_{coef}\,.\,\mathrm{N}\,),\ \text{where:}$$

- I is the number of iterations needed for the optimization algorithm (such as gradient descent) to converge.

- Nb_{coef} is the number of coefficients (n x (n-k+1)).

- N is the number of training samples (2^{n-k}in our case, all possible syndromes).

Regarding our case of using input syndromes and errors with length n as output, the complexity training parameters of LRDec and 2LRDEC in BCH(n,k,t)

code are listed below:

Table 2: Complexity Training Parameters

Parameters	LRDec	2LRDec
NC	n	$n/2$
I	$I = 100000$	$I' = 10000$
N	2^{n-k}	2^{n-k-1}

So, taking all these elements into account, the training complexity becomes:

Table 3: Complexities Training Comparison between 2LRDec and LRDec

Decoder	Training Complexity
LRDEC	$C_1 = O(\mathrm{I}\,.\,n.\,(n-k)\,.\,2^{n-k})$
2LRDEC	$C_2 = O(\mathrm{I}'\,.\,n/2.\,(n-k)\,.\,2^{n-k-1})$

With sample calculation of the 2LRDec's complexity training will be:

$$C_2 = C_1/40$$

So, the use of multi-model in logistic regression, especially 2 models (2LRDec) reduces the training complexity to produce the decoding models for the bigger code length.

5.3.2 Run-Time Complexity:

Let's consider a BCH code $C(n,k,t)$, where n is the code length and t is the error-correcting capability, The space runtime complexity of the logistic regression model, is calculated by:

$$C_{SLR} = O\left(Nb_{coef}\right) = O(n(n-k))$$

This is markedly lower than The space complexity of a hashing-based syndrome decoder like HSDec depends on the number of possible syndromes and corresponding error patterns it has to store. The total number of error patterns that HSDec needs to store for decoding would be the combinations of n things taken t at a time. This is because, in the worst-case scenario, we need to account for all possible combinations of t errors that could occur in the n-length codeword. This can be calculated using binomial coefficients:

$$C(n,t) = n!\,[t!\,(n-t)!]$$

The syndromes that need to be stored are the unique syndromes that correspond to each of these error patterns. Therefore, the total storage or space complexity would also be of the order of $C(n,t) + 2^{n-k}$., the space complexity of HSDEC decoder would be:

$$C_{SHS} = O(n!\,[t!\,(n-t)!] + 2^{n-k})$$

This means that the space requirements will grow exponentially with increasing code length and error-correcting capability.

So, using only 1 or 2 models in LRDec significantly reduces the space complexity compared to a hashing-based decoder like HSDec, especially for larger code lengths. This makes the LRDec approach more scalable and more suitable for applications where space efficiency is a critical concern.

5.4. Discussion:

The 2LRDec splits the classes (correctable errors) and samples (syndromes) into two groups, reducing the complexity of training the logistic regression models by effectively halving the problem size. One primary advantage of this method is that it significantly reduces the computational requirements by utilizing two smaller, more manageable logistic regression models, each handling a fraction of the original problem size. This has the potential to enhance the scalability of the system, particularly for larger codes, where conventional methods can struggle with the increased computational demands. Furthermore, this method also optimizes storage needs. Unlike the HSDEC, which necessitates storing an extensive table of syndromes and errors, the 2LRDec approach only requires the storage of two trained models, effectively decreasing the spatial complexity.

In conclusion, the 2LRDec approach presents a promising alternative for decoding ECCs. Its lower computational complexity and reduced storage requirements make it particularly suitable for applications dealing with larger codes, where the efficiency and scalability of decoding algorithms are of paramount importance. Future work could further

optimize this approach or explore its application to other types of codes.

6. CONCLUSION

In this article, we focused on enhancing our logistic regression-based decoder (LRDec) by employing various techniques to improve the performance of the logistic regression model. As a result, we successfully developed a novel decoder called 2LRDec for linear codes. This decoder has demonstrated significant improvements in bit error rate (BER) performance for BCH codes such as BCH(15,5,7), BCH(15,7,5), BCH(15,11,3), BCH(31,21,5), BCH(31,26,3), BCH(31,16,7), BCH(63,51,3), and BCH(63,45,7). The key innovation behind these improvements is the division of the syndrome basis, which reduces the complexity of the training phase and enables the development of the two models that make up the 2LRDec decoder. This work not only showcases the potential of machine learning-based decoders in the context of linear error-correcting codes but also opens up new avenues for future research. One such direction is the exploration of the m-LRDec decoder, which would consist of m individual LRDec decoders working together to further enhance BER performance for linear codes. This approach could potentially facilitate the development of decoders for more extensive codes with lengths up to 63 bits, broadening the applicability and effectiveness of logistic regression-based decoding techniques. Moreover, the insights gained from this study contribute to a deeper understanding of the factors influencing the performance of machine learning-based decoders and provide valuable guidance for designing more advanced, efficient, and robust error correction solutions for digital communication systems.

REFERENCES:

[1] C. I. Imrane, N. Said, B. E. Mehdi, E. K. A. Seddiq, et M. Abdelaziz, « Machine learning for decoding linear block codes: the case of multi-class logistic regression model », Indones. J. Electr. Eng. Comput. Sci., vol. 24, no 1, Art. no 1, oct. 2021, doi: 10.11591/ijeecs.v24.i1.pp538-547.

[2] S. Nouh, I. Chana, et M. Belkasmi, « Decoding of Block Codes by using Genetic Algorithms and Permutations Set », Int. J. Commun. Netw. Inf. Secur. IJCNIS, vol. 5, no 3, Art. no 3, nov. 2013, doi: 10.54039/ijcnis.v5i3.428.

[3] M. S. el Kasmi Alaoui, S. Nouh, et A. Marzak, « Two New Fast and Efficient Hard Decision DecodersBased on Hash Techniques for Real Time Communication Systems », 2019, p. 448-459. doi: 10.1007/978-3-319-91337-7_40.

[4] E. Nachmani, Y. Be'ery, et ..., « Learning to decode linear codes using deep learning », 2016 54th Annu. Allerton …, 2016, [En ligne]. Available on:https://ieeexplore.ieee.org/abstract/document/7852251/

[5] E. Nachmani, E. Marciano, L. Lugosch, et ..., « Deep learning methods for improved decoding of linear codes », IEEE J. …, 2018, [En ligne]. Available on:https://ieeexplore.ieee.org/abstract/document/8242643/

[6] E. Nachmani, E. Marciano, D. Burshtein, et ..., « RNN decoding of linear block codes », ArXiv Prepr. ArXiv, 2017, [En ligne]. Available on:https://arxiv.org/abs/1702.07560

[7] E. Nachmani, Y. Bachar, E. Marciano, D. Burshtein, et ..., « Near maximum likelihood decoding with deep learning », ArXiv Prepr. ArXiv, 2018, [En ligne]. Available on:https://arxiv.org/abs/1801.02726

[8] L. Lugosch et W. J. Gross, « Neural offset min-sum decoding », 2017 IEEE Int. Symp. …, 2017, [En ligne]. Available on:https://ieeexplore.ieee.org/abstract/document/8006751/

[9] L. Lugosch et W. J. Gross, « Learning from the syndrome », 2018 52nd Asilomar Conf. …, 2018, [En ligne]. Available on:https://ieeexplore.ieee.org/abstract/document/8645388/

[10] A. Bennatan, Y. Choukroun, et ..., « Deep learning for decoding of linear codes-a syndrome-based approach », 2018 IEEE Int. …, 2018, [En ligne]. Available on:https://ieeexplore.ieee.org/abstract/document/8437530/

[11] F. Carpi, C. Häger, M. Martalò, R. Raheli, et H. D. Pfister, « Reinforcement Learning for Channel Coding: Learned Bit-Flipping Decoding », 2019 57th Annu. Allerton Conf. Commun. Control Comput. Allerton, p. 922-929, sept. 2019, doi: 10.1109/ALLERTON.2019.8919799.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[12] Ruan, Xin "Deep Learning Algorithms for BCH Decoding in Satellite Communication". Highlights in Science, Engineering and Technology. 38. 1104-1115, (2023).

[13] Xiangyu Chen, Min Ye, "Neural decoders with permutation invariant structure", Journal of the Franklin Institute, Volume 360, Issue 8,2023, Pages 5481-5503.

[14] D.Khebbou, I.Chana, H.Ben-Azza, "Decoding of the extended Golay code by the simplified successive-cancellation list decoder adapted to multi-kernel polar codes", Telecommunication Computing Electronics and Control, 2023, 21(3), pp. 477–485.

[15] Boualame, H., Belkasmi, M. & Chana, I. "Efficient Decoding Algorithm for Binary Quadratic Residue Codes Using Reduced Permutation Sets". Journal of Computer Science, 19(4), (2023). 526-539. https://doi.org/10.3844/jcssp.2023.526.539

[16] Seddiq El Kasmi Alaoui, Zouhair Chiba, Hamza Faham, Mohammed El Assad, Said Nouh, "Efficiency of two decoders based on hash techniques and syndrome calculation over a Rayleigh channel", International Journal of Electrical and Computer Engineering (IJECE), Vol 13, No 2, April 2023.

[17] Bulso, N.; Marsili, M.; Roudi, Y. On the Complexity of Logistic Regression Models. *Neural Computation* **2019**, *31* (8), 1592–1623. https://doi.org/10.1162/neco_a_01207.