Indonesian Journal of Electrical Engineering and Computer Science
Vol. 24, No. 1, October 2021, pp. 538∼547
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v24.i1.pp538-547

Machine learning for decoding linear block codes: case of multi-class logistic regression model

Chemseddine Idrissi Imrane[1], Nouh Said[2], Bellfkih El Mehdi[3], El Kasmi Alaoui Seddiq[4], Marzak Abdelaziz[5]
[1,2,4,5]LTIM Lab, Faculty of Sciences Ben M'sik, Hassan II University of Casablanca, Morocco
[3]L3A Lab, Faculty of Sciences Ben M'sik, Hassan II University of Casablanca, Morocco

Article Info

Article history:

Received Apr 27, 2021
Revised Aug 13, 2021
Accepted Aug 23, 2021

Keywords:

BCH codes
Error correction codes
Logistic regression
Machine learning
SoftMax
Syndrome decoding

ABSTRACT

Facing the challenge of enormous data sets variety, several machine learning-based algorithms for prediction (e.g, Support vector machine, multi layer perceptron and logistic regression) have been highly proposed and used over the last years in many fields. Error correcting codes (ECCs) are extensively used in practice to protect data against damaged data storage systems and against random errors due to noise effects. In this paper, we will use machine learning methods, especially multi-class logistic regression combined with the famous syndrome decoding algorithm. The main idea behind our decoding method which we call logistic regression decoder (LRDec) is to use the efficient multi-class logistic regression models to find errors from syndromes in linear codes such as bose, ray-chaudhuri and hocquenghem (BCH), and the quadratic residue (QR). Obtained results of the proposed decoder have a significant benefit in terms of bit error rate (BER) for random binary codes. The comparison of our decoder with many competitors proves its power. The proposed decoder has reached a success percentage of 100% for correctable errors in the studied codes.

Corresponding Author:

Chemseddine Idrissi Imrane
LTIM Lab
Hassan II University
Casablanca, Morocco
Email: imran.chems@gmail.com

1. INTRODUCTION

In the last few years, telecommunication technologies have known huge innovations in order to have good speed, and reliable communication between all connected objects, e.g, smartphones, computers, and cameras. The internet of things (IoT) combined to the arrival of 5G in the 3^{rd} generation partnership project [1] have guaranteed very fast speed of data transmission, compared to the older version 4G long term evolution (LTE) [2], the new radio (NR) for 5G adopts a new strategy in error-correction by using transformation in the data channel which uses low density parity check codes (LDPC) and the control channel which uses 2 dimensional tail-biting convolutional codes (TBC) [3]. This technology invests in the hardware and transmission's power to reduce the effect of signal noise. In the same context, some researchers invest in software development, Giuseppe Aceto [4] has conducted a study by using deep learning (DL) for classifying the mobile encrypted traffic. A novel approach has been presented by Tim O'Shea [5] in the design of communication by using DL for the physical layer. In our case, the design of our model is based on the simplified model of communication system, combined with the logistic regression classification in the process of decoding channel Figure 1.

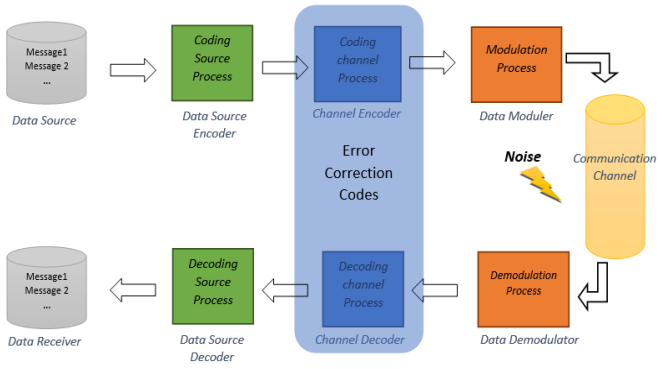

Figure 1. Simplified model of communication systems

Depending on the way used to add redundancy in the messages to be transmitted, it is possible to divide the error correcting codes into two major families: block codes and convolution codes [6]-[7]. The block codes encode data by block independently of other blocks whereas the convolutional codes process the current block not only in relation to the current input of the channel encoder but also in relation to the previous entry blocks [8]. In Figure 2, we present a classification of linear block codes.

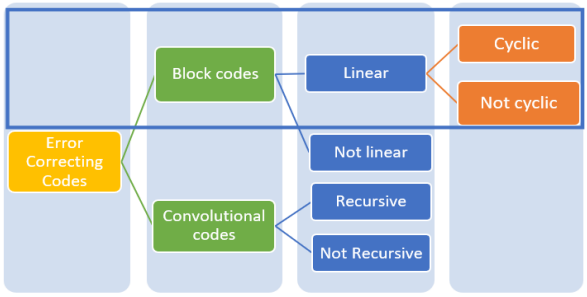

Figure 2. Error correction code classification

The category of linear-block codes is described as a code C(n,k,d); its length is n; its dimension is k, and its minimal distance is d. The particularity of this type of codes is that each linear combination of codewords is another codeword. In this paper, we will work on block codes, namely the category of systematic binary linear block codes. A linear block code C(n, k) is a set comprising 2^k code words so that the linear combinations of the k information bits would generate n bits of each codeword. This property allows to define C by a matrix G called generator matrix or coding matrix which can be used to associate with any information word $m = (m_1, m_2,\ldots, m_k)$ composed of k bits a codeword $c = (c_1, c_2,\ldots, c_n)$ as illustrated in the following Figure 3.

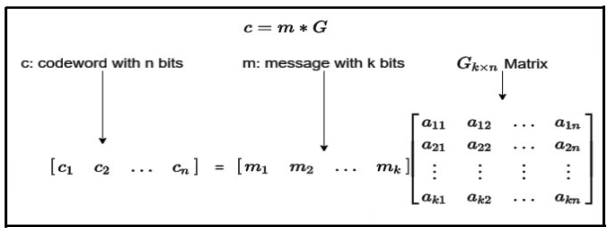

Figure 3. Codewords calculation

The matrix G can be written in systematic form as in (1) where I_k is the identity matrix of order k and P is a binary matrix of order (k,n-k). In this respect, for any code C(n, k), we can calculate a matrix H of order (n-k, n) whose lines are orthogonal to the lines of G, that is to say $G * H^T = 0$. The matrix H is called the

parity matrix (control matrix) of the code C, and the matrix G the generator matrix. The vector space $C^{\perp}$ is a linear code called the dual code C of dimension n-k and of length n. If the code C is generated by systematic generating matrix G, its dual code $C^{\perp}$ can be generated by the matrix H as follows:

$$G = [I_k|P], \quad H = [P^T|I_{n-k}] \tag{1}$$

The BCH code for communication and storage data has been discovered by Hocquenghem [6]. It was developed by Bose and Ray-Chaudhuri [7], it is one of most famous codes of powerful error capability [8]. The technical specification of BCH codes is as follows:

$$\begin{cases} n = 2^m - 1, \forall m \geq 3, & \text{n: block length} \\ k \leq n - mt, & \text{k: number of message bits} \\ d \geq 2t + 1, & \text{d: the minimum distance, t: the designed error correcting capability} \end{cases} \tag{2}$$

In this paper, we will first expose a general literature review of methods and approaches in error correcting codes. Then, we will develop our new decoder LRDec based on logistic regression model which we think is one of the most efficient machine learning algorithm. We should inform you that we have combined our LRDec with the syndrome decoding technique so as to eliminate the decoding syndrome motive table. After that, we will expose different obtained results for linear codes. Finally, we will compare and discuss them with other existing decoders' results (e,g: HSDEC [9]-[10], ARdecGA [11] and BERT [12]). In addition to this, in contrast to other decoders, our LRDec is has been successfully applied to another kind of linear codes which is named quadratic residual code (QR). Hence, the comparison between our LRdec and HSdec in QR codes has shown a significant efficiency in terms of BER.

The sequel of this paper is organised as follows: In section 1, we will present some related works. In section 2, an overview of machine learning techniques, especially the logistic regression models method will be detailed, and we will develop the proposed framework of the LRDec. In section 3, we will expose our results and discussion. At the end of this paper, we will present our conclusion and suggest some possible future directions of this work.

2. RELATED WORKS

Being aware of the difficulties of ECC problem, many decoders are used to enhance and improve the reliability and performance in terms of bit error rate (BER). Figure 4 presents the main classes of decoding techniques. Some decoders are based on algebraic theory such as the algorithms developed through solving nonlinear multivariate equations obtained from the identities of Newton [13]-[14], the Berlekamp-Massey algorithm [15] which is based on the calculation of syndromes and the definition of an error locator polynomial, the algorithm of Chase [16] and the algorithm of Hartmann Rudolf [17]-[18].

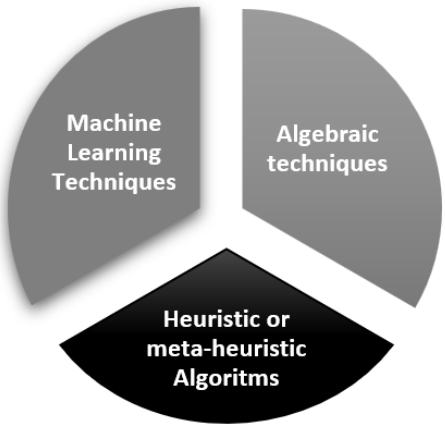

Figure 4. Decoding algorithms classification

However, some algebraic techniques, aforementioned, require a large wide of computational operations, in terms of sum and product, in the used finite field. This time complexity makes their implementation in real time systems very hard. That is why, the algorithms with fast detection and correction are strictly required here. To solve this problem, several researches have been carried out to develop both heuristic or meta-heuristic algorithms and machine learning techniques which aim to detect and correct transmission occurred errors with high accuracy and high speed.

There are other methods that use non-algebraic techniques, such as genetic algorithms that belong to evolutionary techniques [18]-[19]. We have also found a work that uses local search to find errors [20]-[21]. Moreover, several other works use hashing techniques [22]-[23]. In this regard, the decoder named HSDec [9] based on syndrome calculation and hash techniques has significantly contributed to accelerate the search time of the error vector in the pattern error table. Also, we have found many works that deal with error correction by involving the deep learning algorithms [24]. The authors of these works exploit the properties of linear codes with the functionalities of deep learning algorithms to develop a decoder of BCH codes. Another decoder, applicable to polar codes, based on deep neural networks has proved to be a very efficient polar decoder [25]. A deep learning algorithm to ameliorate and improve the belief propagation algorithm [26] is applicable to BCH codes. The researchers exploited a machine learning algorithm combined with the syndrome calculation to improve the decoder performance in terms of BER and time complexity; this method is applicable to linear block codes [27].

3. THE PROPOSED MODEL

In this section, we will firstly give a brief summary of the techniques of machine learning. Secondly, we will present the design of our decoder LRDec as well as the technical specifications of its implementation.

3.1. Machine learning models

Machine learning is an extensively employed domain in artificial intelligence that is working on analyzing and exploring features of any kind of data, including nominal data, with the purpose to generate models and make an accurate future prediction. The latter can be presented by the way of functions construction from the data X=($X_0, X_1, X_2, .., X_j, ..$) to predict the values of Y. This is illustrated in Figure 5.

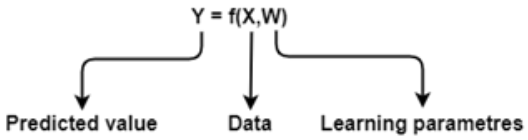

Figure 5. Equation 1

If the type of Y is discrete values, we talk about classification learning. But, if the type of Y is a continuous numeric value, the learning is regression type. For the classification problem, there is a set of algorithms in machine learning, such as K-nearest neighbors (KNN) algorithm,support vector machine (SVM) algorithm and logistic regression (LR) algorithm that can be utilized to produce very high classification accuracies. The most prevalent approach for evaluating binary response data is logistic regression. The logistic regression model estimates the likelihood of output Y as a function of one or more predictor X_j. Despite its name, the logistic regression model is a classification model, rather than a regression model. It is a simple and powerful method to solve problems with binary and linear classification. In this paper, we will use the logistic regression, and namely, the multi-class logistic regression model.

In a classification problem, the objective is that the probability of the correct class Y to which the inputs x_i belong must be maximized. We train the logistic regression model to find the appropriate weights, which will enable us to calculate the probability of each input belonging to the class Y. The Sigmoid function will always be used in this case. All these specifications are presented in Figure 6.

In the same way, as shown in Figure 7, we calculate the values $f(z_i)/i = 1, \ldots, k$, for k classes. Then, we compare these values with each other to find the target class. The values should be normalised so as to be considered as probability [28]. For this purpose, the softmax function is used.

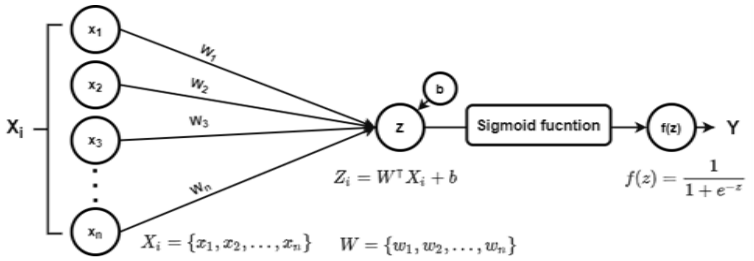

Figure 6. Logistic regression model

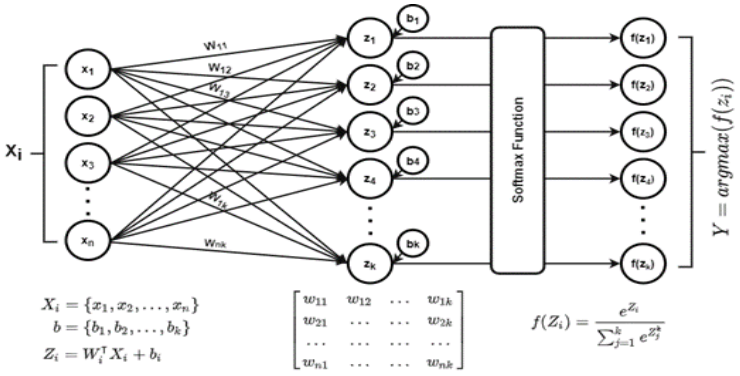

Figure 7. Multi-class logistic regression model

3.2. The proposed decoding method based on logistic regression model

In this section, we will present our proposed model named LRDec for linear bloc codes. In this regard, we will work on BCH and Quadratic Residue codes. This decoder can be generalized to any linear block code defined by its generator matrix or its generator polynomial. This new LRDec based on the logistic regression will be compared to many famous decoders (HSDEC, ARDecGA and BERT) in terms of BER performance and complexity. LRDec is designed to improve syndrome decoding technique. In Figure 8 we present the architecture of the LRDec. On the one side, The features inputs S_i are the n-k bits of the received word's syndrome. On the other side, the outputs Y_i are the classes of all the different correctable errors converted to decimal values.

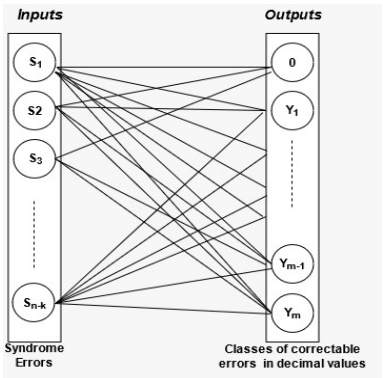

Figure 8. LRDec model

3.2.1. The data pre-processing

Before any model creation procedure, i.e. the training phase, it is essential to prepare the input data as well as the output data. For the output, the aim is to define specific classes for each correctable error. As for the input, the objective is to generate all possible syndrome vectors with length equal to n-k.

a- Outputs Y

We start by generating all possible errors of length n presented by the binary vector with weights (number of 1 in each vector) less than or equal to the error correcting capability of the code. After that, we have to convert all the generated vectors to decimal values. Finally, we group all these decimal values to list Y_1. This list represents the classes for all correctable errors. As for the incorrigible errors, we are going to add $Y_2 = 0$, to keep the received word when it 's impossible to correct.

$$\begin{cases} Y = [0, y_1, \ldots, y_m] \quad | \quad m = \sum_{i=0}^{t} C_n^i \\ Y_1 = [y_i = decimal(err_i) \quad | \quad 0 \leq weight(err_i) \leq t] \\ Y_2 = [0 \quad | \quad t < weight(err_i)] \end{cases} \tag{3}$$

b- Inputs X

After having created the list of classes Y with Y_1 for the correctable errors and Y_2 for incorrigible errors, we will create another list X by generating all syndromes of errors. After this process we will assign labels for each x_i in X. The correctable errors Binary(y_i) must have a label $x_i = syndrome(ConvertBin(y_i))$; and for the other x_j the corresponding label is Y_2=0.

$$\begin{cases} X = [x_0, x_1, \ldots, x_p] \quad | \quad p = 2^{n-k} \\ X_1 = [x_i = Syndrome(err_i) \quad | \quad 0 \leq i \leq m] \quad \rightarrow label - c_i = y_i = Decimal(err_i) \\ X_2 = [All - syndromes - not - in - X_1] \quad \rightarrow labels - c_0 = Y_2 = 0 \end{cases} \tag{4}$$

In general, the database will be formed by $p = 2^{n-k}$ samples and $m = \sum_{i=0}^{t} C_i^n$ classes:

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_p \end{bmatrix} = \begin{bmatrix} S_{11}, & S_{12}, & \ldots & S_{1n-k} \\ S_{21}, & S_{22}, & \ldots & S_{2n-k} \\ \vdots & \vdots & \vdots & \vdots \\ S_{p1}, & S_{p2}, & \ldots & S_{pn-k} \end{bmatrix} \begin{Bmatrix} \text{Syndrome of correctable error 1} \\ \text{Syndrome of correctable error 2} \\ \vdots \\ \text{Syndrome of correctable error m} \\ \text{Syndrome of incorrigible error m+1} \\ \vdots \\ \text{Syndrome of incorrigible error p} \end{Bmatrix} \xrightarrow[Labels]{} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Example: BCH (7, 4, 3)

In this code, the first step is to generate all the possible correctable errors Err_i, the second step is to calculate all the possible syndromes x_i by calculating the product between errors and the matrix of the parity check H for BCH (7, 4, 3). In Table 1, we show the results of the generating data-set for the BCH (7, 4, 3) code. In Table 2, we show the number of classes *Card(Y)*=$\sum_{i=0}^{t} C_n^i, and samples$*Card(X)*=$2^{n-k}$ for creating LRDec model.

Table 1. Data-set (X;Y) for BCH(7;4;3)

-	Err_i	x_i=syndrome(Err_i)	y_i
0	$Err_0 = [0000000]$	[0, 0, 0]	0
1	$Err_1 = [0000001]$	[0, 0, 1]	1
2	$Err_2 = [0000010]$	[0, 1, 0]	2
3	$Err_3 = [0000100]$	[1, 0, 0]	4
4	$Err_4 = [0001000]$	[1, 1, 1]	8
5	$Err_5 = [0010000]$	[0, 1, 1]	16
6	$Err_6 = [0100000]$	[1, 0, 1]	32
7	$Err_7 = [1000000]$	[1, 1, 0]	64

$$x_i = Syndrome(Err_i) = err_i.H^T, \quad \text{Where} \quad H = \begin{bmatrix} 1101100 \\ 1011010 \\ 0111001 \end{bmatrix}$$

Table 2. Number of classes and samples

BCH(n,k,d)	Card(X)	Card(Y)
BCH(7,4,3)	8	8
BCH(15,7,5)	256	121
BCH(15,5,7)	1024	576
BCH(31,16,7)	32768	4992
BCH(31,21,5)	1024	4992

3.2.2. The training process

Once the data-set (X; Y) is prepared, the next process is training the Logistic regression model, knowing that the machine learning paradigm discussion between over-fitting and under-fitting is not necessarily included in our strategy because the data-set presents all possible values. In this case, having a training error equal to 100% does not mean that we have the over-fitting case. However, it is our main objective to have a model which corrects all possible errors of weights $\leq$ t. In Table 3, we show that our decoder correct all errors of weight less than or equal to the error correcting capability of the studied codes. Thus the percentage success of LRDec on these codes is 100% for correctable errors.

Table 3. The percentage success of LRDec

Code BCH(n,k,d)	Error correcting capability	% success of LRDec on correctable errors
BCH(15,7,5)	1	100%
BCH(15,5,7)	3	100%
BCH(31,16,7)	3	100%
BCH(31,21,5)	2	100%

4. RESULTS AND DISCUSSION

In order to show the huge success of our LRDec, we are going to plot its performances (for some linear BCH codes) in terms of bit error rate (BER) for different values of signal noise ratio (SNR), in the additive white gaussian noise channel (AWGN) and with binary phase shift keying (BPSK) modulation. The simulation parameters are displayed in the following Table 4:

Table 4. Default simulation parameters

Simulation parameters	values
Channel	AWGN
Modulation	BPSK
Minimum number of residual bit in errors	200
Minimum number of transmitted blocks	10000

4.1. Results

In general, in AWGN channel transmission without coding decoding algorithms, we have a value of BER=10^{-5} for SNR=9.6 dB. The result obtained in Figure 9 for the LRDec in BCH (15, 5, 7), BCH (15, 7, 5) and BCH (15, 11, 3) has shown that we have gained a decoding approximate to 1.2 dB and 0.8 dB. In Figure 10, with the BCH (31, 21, 5) and BCH (31, 16, 7), we have obtained a decoding gain of about 2 dB.

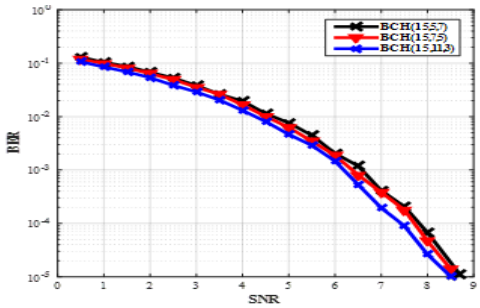

Figure 9. LRDec for some BCH codes with n=15

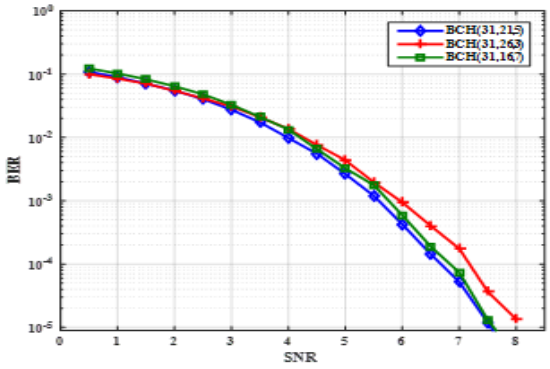

Figure 10. LRDec for some BCH codes with n=31

4.2. Comparison and discussion

To show the efficiency of our decoding model, we have compared its performances with other existing decoders. In Figure 11, we present the performances of the LRDec, ARDecGA [11] and BERT [12] decoders for the BCH code (15, 7, 5). Thanks to the comparisons listed above, the LRDec for the code BCH (15, 7, 5) has proved to be the best decoder in terms of performance and efficiency. For the same purpose, as shown in Figure 12, the LRDec and the HSDec [9] have the same performance in the code BCH (31, 16, 7). What is more, in Figure 13, we have compared the performance of LRDec with HSDec for a new class of binary code, the QR code (23, 12, 7), and the results have shown that the RLDec is more efficient than HSDec.

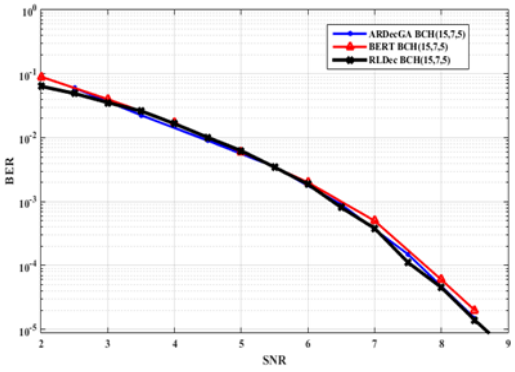

Figure 11. Performance comparison of ARDec, BERT and LRDec for the BCH code (15, 7, 5)

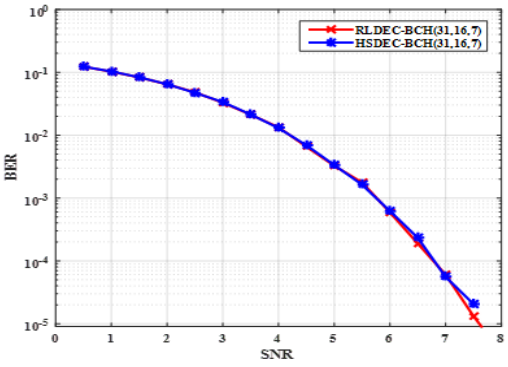

Figure 12. Performance comparison of LRDec and HSDec for the BCH code (31, 16, 7)

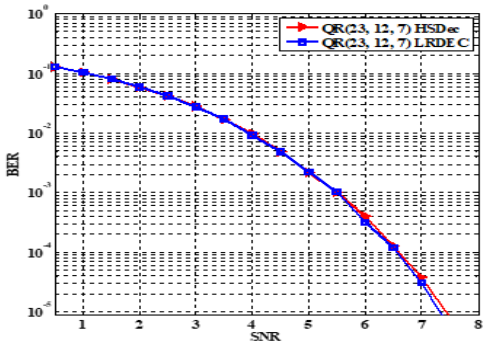

Figure 13. Performance comparison of LRDec and HSDec for the QR code (23, 12, 7)

5. CONCLUSION

In this study, we have focused on the possibility of using a logistic regression model in error correcting codes, especially in syndrome decoding technique applied to linear codes. To achieve this objective, we have successfully designed a new decoder named LRDec with 100% accuracy of training model. The idea behind this choice of methodology is based on the performances guaranteed by machine learning algorithms. Unlike the classical syndrome decoding method, this new decoder LRDec does not need a large syndrome pattern's table. The proposed decoder LRDec has shown significant efficiency in term of BER and has reached the 100% correction of all correctable errors in the studied codes. We have also conducted a comparison between LRDec with three decoders: BERT, HSDec and ARDecGA for decoding BCH (15, 7, 5), BCH (31, 16, 7), and QR (23, 12, 7) codes. The obtained results have shown the success of our proposed model decoder in learning how to calculate directly error from syndrome without using the large table of syndrome decoding process. In perspectives, we plan to study other machine learning models for decoding other non linear codes, and to search about the optimisation of the model by reducing its complexity in terms of the activation function or by discussing the model's parameters in order to improve its efficiency.

REFERENCES

[1] S. Antipoli, "Multiplexing and channel coding," In *3rd Generation Partnership Project*, France. TS 38.212, v15.0.0, Release 15, 3GPP, 2018, pp. 1-100. [Online]. Avaliable: http://www.etsi.org/standards-search.

[2] ETSI, "LTE, evolved universal terrestrial radio access (EUTRA), multiplexing and channel coding," *European Telecommunications Standards Inst,* Sophia-Antipolis, France. TS 136 212 v12.2.0, Release 12, 2014. [Online]. Avaliable: http://www.etsi.org/standards-search.

[3] D. Hui, S. Sandberg, Y. Blankenship, M. Andersson, and L. Grosjean, "Channel Coding in 5G New Radio: A Tutorial Overview and Performance Comparison with 4G LTE," *IEEE Vehicular Technology Magazine*, vol. 13, no. 4, pp. 60-69, 2018, doi: 10.1109/MVT.2018.2867640.

[4] G. Aceto, D. Ciuonzo, A. Montieri and A. Pescapé, "Mobile Encrypted Traffic Classification Using Deep Learning: Experimental Evaluation, Lessons Learned, and Challenges," In *IEEE Transactions on Network and Service Management*, vol. 16, no. 2, pp. 445-458, Jun. 2019, doi: 10.1109/TNSM.2019.2899085.

[5] T. O'shea, and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Transactions on Cognitive Communications and Networking*, vol. 3, no. 4, pp. 563 - 575, 2017, doi: 10.1109/TCCN.2017.2758370.

[6] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffres*, vol. 2, pp. 147-156, Sep. 1959.

[7] R. C. Bose, and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Inf. Control*, vol. 3, no. 1, pp. 68-79, 1960, doi: 10.1016/S0019-9958(60)90287-4.

[8] K. Lee, H. G. Kang, J. I. Park, and H. Lee, "A high-speed low-complexity concatenated BCH decoder architecture for 100 Gb/s optical communications," *J. Signal Process. Syst.*, vol. 66, no. 1, 2012, doi: 10.1007/s11265-010-0519-0.

[9] M. S. E. K. Alaoui, S. Nouh, and A. Marzak, "Two New Fast and Efficient Hard Decision Decoders Based on Hash Techniques for Real Time Communication Systems," In *First International Conference on Real Time Intelligent Systems*, 2019, pp. 448-459, doi: 10.1007/978-3-319-91337-7-40.

[10] M. S. E. K. Alaoui, S. Nouh, and A. Marzak, "High Speed Soft Decision Decoding of Linear Codes Based on Hash and Syndrome Decoding,", *International Journal of Intelligent Engineering and Systems*, vol. 12, no. 1, pp. 94-103, 2019, doi: 10.22266/ijies2019.0228.10.

[11] S. Nouh, I. Chana, and M. Belkasmi, "Decoding of block codes by using genetic algorithms and permutations set," *International Journal of Communication Networks and Information Security*, vol. 5, no. 3, pp. 201-209, 2013.
[12] E. Mohamed, H. Abdelkader, O. Mouhib, and E. I. El Habti, "Performance Study of BCH Error Correcting Codes Using the Bit Error Rate Term BER," *Int. J. Eng. Res. Appl.*, vol. 7, no. 2, pp. 52-54, 2017, doi: 10.9790/9622-0702025254.
[13] I. S. Reed, X. Yin, T. K. Truong, and J. K. Holmes., "Decoding the (24,12,8) Golay code," *IEE Proc. E Comput. Digit. Tech.*, vol. 137, no 3, pp. 202-206, 1990, doi: 10.1049/ip-e.1990.0025.
[14] I. S. Reed, X. Yin, and T. K. Truong, "Algebraic decoding of the (32, 16, 8) quadratic residue code," *IEEE Trans. Inf. Theory*, vol. 36, no 4, pp. 876-880, 1990, doi: 10.1109/18.53750.
[15] E. Berlekamp, R. McEliece, and H. Van Tilborg, "On the inherent intractability of certain coding problems (Corresp.)," *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384-386, 1978, doi: 10.1109/TIT.1978.1055873.
[16] D. Chase, "Class of algorithms for decoding block codes with channel measurement information," *IEEE Trans. Inf. Theory*, vol. 18, no 1, pp. 170-182, 1972, doi: 10.1109/TIT.1972.1054746.
[17] C. Hartmann, and L. Rudolph, "An optimum symbol-by-symbol decoding rule for linear codes," *IEEE Trans. Inf. Theory*, vol. 22, no 5, pp. 514-517, 1976, doi: 10.1109/TIT.1976.1055617.
[18] M. S. E. K. Alaoui, and S. Nouh, "Decoding Algorithm by Cooperation Between Hartmann Rudolph Algorithm and a Decoder Based on Syndrome and Hash," *International Journal of Natural Computing Research (IJNCR)*, vol. 10, no. 1, pp. 15-27, 2021, doi: 10.4018/IJNCR.2021010102.
[19] A. Berkani, A. Azouaoui, M. Belkasmi, and B. Aylaj, "Improved Decoding of linear Block Codes using compact Genetic Algorithms with larger tournament size," *IJCSI International Journal of Computer Science Issues*, vol. 14, no. 1, 2017, doi: 10.20943/01201701.1524.
[20] H. Faham, M. S. E. K. Alaoui, S. Nouh, and M. Azzouazi, "An efficient combination between Berlekamp-Massey and Hartmann Rudolph algorithms to decode BCH codes," *Periodicals of Engineering and Natural Sciences (PEN)*, vol. 6, no. 2, pp. 365-372, doi: 10.21533/pen.v6i2.540.
[21] M. Esmaeili, A. Alampour, and T. A. Gulliver, "Decoding Binary Linear Block Codes Using Local Search," *IEEE Trans. Commun.*, vol. 61, no 6, pp. 2138-2145, 2013, doi: 10.1109/TCOMM.2013.041113.120057.
[22] Y.-H. Chen, C.-F. Huang, and J. Chang, Decoding of binary quadratic residue codes with hash table, IET Commun., vol. 10, no 1, p. 122-130, janv. 2016, doi: 10.1049/iet-com.2015.0546.
[23] C. F. Huang, W. R. Cheng, and C. C. Yu, "A Novel Approach to the Quadratic Residue Code," In *Advances in Intelligent Information Hiding and Multimedia Signal Processing*, 2017, pp. 187-194, doi: 10.1007/978-3-319-50212-0-23.
[24] I. Be'Ery, N. Raviv, T. Raviv, and Y. Be'Ery, "Active Deep Decoding of Linear Codes," *IEEE Trans. Commun.*, vol. 68, no 2, pp. 728-736, 2020, doi: 10.1109/TCOMM.2019.2955724.
[25] W. Xu, Z. Wu, Y. Ueng, X. You, and C. Zhang, "Improved polar decoder based on deep learning," In *2017 IEEE International Workshop on Signal Processing Systems (SiPS)*, 2017, pp. 1-6, doi:10.1109/SiPS.2017.8109997.
[26] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, 2016, pp. 341-346, doi: 10.1109/ALLERTON.2016.7852251.
[27] L. P. Lugosch, "Learning Algorithms for Error Correction," Master of Thesis, McGill University, Montréal, Canada, 2018.
[28] H. Matsui, "Variable and boundary selection for functional data via multiclass logistic regression modeling," *Computational Statistics and Data Analysis*, vol. 78, pp. 176-185, 2014, doi: 10.1016/j.csda.2014.04.015.