A new efficient decoder of linear block codes based on ensemble learning methods

Mohammed El Assad[1], Said Nouh[1], Imrane Chemseddine Idrissi[1], Seddiq El Kasmi Alaoui[2], Bouchaib Aylaj[3], Mohamed Azzouazi[1]
[1]Laboratory of information technologies and modeling (LTIM), Faculty of Sciences Ben M'sick, Hassan II University, Casablanca, Morocco
[2]Laboratorium Information System (LIS), Faculty of Sciences Ain Chock, Hassan II University, Casablanca, Morocco
[3]Department of Computer Sciences, CRMEF, Rabat, Morocco

Article Info

Article history:

Received Aug 2, 2023
Revised Oct 12, 2023
Accepted Nov 15, 2023

Keywords:

BCH codes
Boosting
Ensemble learning methods
Error correcting codes
Quadratic residue codes
Syndrome decoding

ABSTRACT

Error-correcting codes are used to partially or completely correct errors as much as possible, while ensuring high transmission speeds. Several machine learning models such as logistic regression and decision tree have been applied to correct transmission errors. Among the most powerful machine learning techniques are aggregation methods which have yielded to excellent results in many areas of research. It is this excellence that has prompted us to consider their application for the hard decoding problem. In this sense, we have successfully designed, tested and validated our proposed EL-BoostDec decoder (hard decision decoder based on ensemble learning-boosting technique) which is based on computing of the syndrome of the received word and on using ensemble learning techniques to find the corresponding corrigible error. The obtained results with EL-BoostDec are very encouraging in terms of the binary error rate (BER) that it offers. Practically EL-BoostDec has succeed to correct 100% of errors that have weights less than or equal to the correction capability of studied codes. The comparison of EL-BoostDec with many competitors proves its power. A study of parameters which impact on EL-BoostDec performances has been established to obtain a good BER with minimum run time complexity.

Corresponding Author:

Mohammed El Assad
LTIM Lab, Faculty of Sciences Ben M'sick, Hassan II University
Casablanca, Morocco
Email: mohammed.elassad-etu@etu.univh2c.ma

1. INTRODUCTION

The growing exchange and transmission of data in our society necessitates the implementation of specialized processes to detect and correct errors that may arise during communication via various channels. Whether it's through wired connections or wireless networks, ensuring the reliable transmission of digital information is crucial. Regardless of the specific type of transmission medium being used, the primary focus lies in establishing robust mechanisms that can handle error detection and correction, thereby maintaining the integrity of the transmitted data.

The information can be of any type provided that it can be given a digital representation: texts, images, sounds, and videos. The transmission of these types of data is ubiquitous in all systems related to data processing and especially in the world of telecommunications. The latter is quite often parasitized; however, it is essential that the information collected or transmitted is well received. There is therefore a need to "make the transmission more reliable": this is the role of error correcting codes. During the transmission of the message between the sender and the receiver, it undergoes several operations including coding and decoding.

In this article we focus our study on linear block codes. Each message to be processed (transmit and store) is segmented into a set of blocks of k elements, where each block is coded by the channel coder in such a way as to transform it into a block of n elements (n>k). In the rest of this paper we represent a code C by C(n, k, d) where n, k, and d are respectively the dimension, the length and the minimum distance of the C code. G is a systematic generator matrix of C and H a parity check matrix of C.

Within the realm of linear error-correcting codes, there exists a notable subgroup known as cyclic codes. Unlike other linear codes, which are defined by generator matrices, cyclic codes are characterized by generator polynomials. This distinction simplifies the encoding process, making it more efficient and suitable for various applications. Two well-known instances of cyclic codes are BCH codes, named after their creators Bose, Ray-Chaudhuri, and Hocquenghem, and quadratic residue (QR) codes. Both BCH and QR codes leverage the cyclic properties to enhance error correction and data storage capabilities in diverse fields, such as data transmission and storage.

Decoding an error-correcting code is an nondeterministic polynomial time (NP)-hard problem [1], [2]. Hard decision decoders work on the binary form of outputs of the transmission channel. In this article we focus our work on the application of an artificial intelligence-based model for the decoding problem. Given the complexity of the issue, numerous linear code decoding techniques have been developed. These include algorithms developed by solving multivariate nonlinear equations derived from Newton's identities [3]–[5]. Chien [6] offer methods for deciphering binary systematic QR codes using lookup tables. The one-to-one correlation between syndromes and correctable error patterns serves as the foundation for the decoding technique. Without the requirement for operations like addition and multiplication over a limited field, the technique uses lookup tables to directly find errors. They also discuss ways to use shift-search decoding to lower memory needs.

Other approaches make use of local search and genetic algorithms. Some articles [7], [8] present some learning-based algorithms for error correction. A new deep-learning technique for enhancing the belief propagation (BP) algorithm for decoding linear block codes is presented in [7]. Imrane *et al.* [8] used a machine learning approach along with a syndrome calculation to enhance the performance of another BP-based technique, which is applicable to BCH and QR codes, in terms of bit error rate (BER) and time complexity. In the same spirit, Nachmani *et al.* [9] have presented an architecture for recurrent neural networks that can correct errors efficiently. In spite of the huge example space, it has been discovered that employing a feed-forward neural network design can outperform classical BP decoding. Alaoui *et al.* [10] have used hash techniques in conjunction with syndrome computation to create decoders with shorter run times. Their suggested decoders work with linear codes. Chu *et al.* [11] presents an efficient algorithms to reduce the number of queries for the guessing random additive noise decoding (GRAND) when the codes are systematic and cyclic. The artificial reliabilities based decoding algorithm by using genetic algorithms (ARDecGA) decoder is described in [12]. It computes an artificial reliability vector for the binary word received and uses a genetic algorithm to locate the binary word with the highest likelihood at this vector. It creates a vector of artificial reliabilities from the binary received word for its decoding procedure.

Boualame *et al.* [13] have presented a solution for decoding the QR(17, 9, 5) code. They propose a methodology that involves identifying the positions of errors within the code. Specifically, they utilize the inverse free Berlekamp-Massey algorithm [14] to decode the code by determining the error-locators of algebraic-geometric codes. This approach offers a systematic way to decode the QR(17, 9, 5) code and retrieve the encoded information accurately. According to Niharmine *et al.* [15], a novel soft decoding method based on the simulated annealing (SA) algorithm is presented. The decoder's key contribution is that it provides nearby solutions based on the received codeword's most accurate information as the starting solution. By minimizing the search space and taking into account the error-correcting capability of the code, the performances that they obtained are enhanced.

Joundan *et al*. [16] presented an evolutionary algorithm to design good linear codes with large minimum weight and low dual minimum distance. Certain codes obtained using their method are the best in terms of the minimum height distance they provide. For instance, the four codes listed in Table 1 have the lowest distance that can be accommodated given their lengths and dimensions. SA is utilized to correct many errors [17]. Many other decoders are developed to enhance correcting quality. Alaoui *et al.* [18] have studied the efficiency of their decoders over a Rayleigh channel. Ruan [19] present general insights on applying neural network decoders to satellite communications. Chen and Ye [20] proposed a neural decoder. Khebbou *et al.* [21] have adapted a polar code decoding technique in favor of the extended Golay code. Boualame *et al.* [22] have proposed a decoder that uses a condensed set of permutations drawn from the huge automorphism group of QR codes to rectify t or fewer incorrect bits in the received word.

Table 1. Some optimal codes constructed by Joundan *et al*. [16]

Code	n	k	d	Lower bound	Header generator
J(26, 13, 7)	26	13	7	7	1010110000110
J(28, 21, 4)	28	21	4	4	111010101101110000111
J(48, 32, 6)	48	32	6	6	10110001100011111011111001101011
J(52, 39, 6)	52	39	6	6	101000001111100010011001011011100001101

In the realm of machine learning, computers have the ability to learn and evolve through experience, without the need for explicit programming [23]. These machine learning models employ various approaches to analyze and learn from data in order to make accurate predictions. To improve the accuracy and reliability of predictions, ensemble methods are utilized, which combine the predictions of multiple predictors. Ensemble learning encompasses different families of methods such as boosting, bagging, and stacking, each with its own unique characteristics and advantages. In general, there are many families of methods like:

- Adaptatives methods (boosting) where the parameters are iteratively adapted to produce a better mixture. Many weak learners learn sequentially and their decisions are combined following a deterministic strategy. In this paper, we focus our work on this type of ensemble learning.
- Averaging methods (bagging, random forest) where many strong learners learn independently from each other in parallel and their decisions are combined following some kind of deterministic averaging process.
- Stacking that use a meta-model to output a prediction.

The principle of boosting: is to evaluate a sequence of weak learners on several slightly modified versions of the training data. The decisions obtained are then combined by a weighted sum to obtain the final model. Decorrelated weak classifiers can be generated by iteratively learning the classifiers and by modifying the training sample at each iteration. The importance of well ranked examples decreases. The importance of poorly classified examples increases. Obtained classifiers can be combined by computing the weighted sum of decisions. There are many boosting algorithms. The best known is the adaptive boosting (AdaBoost) [24].

The AdaBoost algorithm: is a powerful machine learning algorithm that employs the concept of combining weak classifiers to construct a robust classifier. This Algorithm 1 functions by iteratively adjusting the weights of incorrectly classified instances. It assigns higher weights to misclassified examples in each iteration, prompting subsequent weak classifiers to prioritize those instances, thus improving the overall accuracy of the final classifier. The iterative nature of AdaBoost results in a strong classifier that excels at handling complex and challenging classification tasks.

Algorithm 1. The steps and process of AdaBoost algorithm
```
1. N ← the number of training samples
2. Weights ←  [1/N, 1/N, ..............., 1/N] (N times)
3. Alpha_Vector ← empty list
4. For each classifier C :
    a) errors ← [0]*N
    b) for i=1 to N :
            if (the i^th sample is misclassified by C) then : errors [i] ←  1
    c) e ← sum of the weights corresponding to misclassified samples
    d) $\alpha \leftarrow 0.5 * \ln(\frac{1-e}{e})$
    e) w ← [0, 0, 0, ….., 0] (N times)
    f) for i=1 to N :
            if (errors[i] = 1 ) then
               w[i] ← Weights[i] * exp(α)
            otherwise:
               w[i] ← Weight [i] * exp(- α)
       endif
    g) $\text{Weights} \leftarrow \frac{w}{\text{sum}(w)}$
    h) Add the coefficient α to the vector Alpha_Vector
```

Boosting with scikit-learn: it is the AdaBoost classifier class that implements this algorithm. The most important parameters used in this paper are as follows. i) n_estimators: integer, optional (default=10), the number of weak classifiers; ii) learning_rate: controls the speed of change of the weights per iteration; and iii) base_estimator: (default=decision tree classifier) the weak classifier used.

The rest of this paper is organized as follows: section 2 presents the proposed EL-BoostDec decoder (hard decision decoder based on ensemble learning-boosting technique). In section 3 give some simulation results of EL-BoostDec, their interpretations and make a comparison of the proposed decoder with some competitors and we will discuss its power. Finally, last section presents a conclusion and perspectives.

2. METHOD

Presenting our decoder, EL-BoostDec, it functions as a robust hard decision decoder, leveraging the power of ensemble learning and the boosting technique. The decoding process involves the computation of syndromes, and the application of ensemble learning methods aids in the identification and correction of errors within the received data. The preparation and operation of EL-BoostDec are executed in a systematic manner, ensuring efficiency and accuracy in error correction. Our EL-BoostDec works as follows:

– Step 1: the preparation of the dataset containing the attributes that characterize different syndromes (the n-k columns) and the classes that represent the errors, each error will be represented by an integer that is the decimal version of the binary error vector encoded on n bits. In total, our dataset contains n-k+1 columns, the last of them represents the error. When a syndrome does not correspond to any correctable error, of weight lower than or equal to the capability of correction of the code, the null error is attributed to it. In the following, X represents the first n-k columns and Y will represent the last column which is the error column in decimal format.
– Step 2: training a powerful EL-BoostDec classifier (an efficient machine learning model) based on boosting methods to learn to find the error from the syndrome.
– Step 3: using the trained classifier to correct the data transmission errors.

Once the EL-BoostDec classifier has undergone training, its functionality aligns with the algorithm described in the subsequent section, namely Algorithm 2. In this operational phase, the decoder applies the acquired knowledge from the training process to effectively identify and correct errors in the received data. This approach ensures a streamlined and optimized decoding process, showcasing the practical implementation and effectiveness of EL-BoostDec in error correction tasks.

Algorithm 2. The steps and process of EL-BoostDec proposed decoder

```
Input:
    ✓ b : Binary version of the received sequence
    ✓ H : Control matrix
    ✓ EL-BoostDec: Well trained classifier
Output:
    ✓ c : decoded codeword.
Begin
    ✓ S← Compute the syndrome of b : $S=b.H^T$
    ✓ v ← EL − BoostDec (S): prediction of the error v that corresponds to S
    ✓ e ← binary version of the number v
    ✓ c ← b $\oplus$e
End
```

2.1. Implementation of EL-BoostDec

For the practical realization of our decoder, we have chosen to implement it using the scikit-learn package in the Python language. Specifically, we employed the AdaBoost classifier class within scikit-learn to bring the EL-BoostDec algorithm to life. This classifier facilitates the integration of the boosting technique into our decoder. As for the crucial parameters, they play a pivotal role in fine-tuning the decoder's performance, ensuring optimal results during the error identification and correction process:

– n_estimators: the number of weak classifiers used, it varies depending on the code studied, by default its value equal to 5 in this dissertation, except in the case where this parameter is varied to study its impact.
– learning_rate: controls the speed of change of the weights per iteration, by default equal to 0.7 for all the results mentioned in this work except in the case where this parameter is varied for the study of its impact on the performances in section 3.
– base_estimator: by default equal to tree. Decision tree classifier (max_depth=5) for all the results mentioned in this work except the case where this parameter is varied for the study of its impact on the performances in section 3. The parameter max_depth is represented as the threshold on the maximum depth of the tree.

2.2. Construction of dataset

For a binary linear code C(n, k, d) there are 2^{n-k} syndromes that do not all correspond to correctable errors i.e. there are syndromes that correspond to uncorrectable errors whose error that will be predicted as zero. The number of correctable errors (NEC), depends on the error capability of the code t: $\mathrm{NEC} = 1 + \sum_{i=1}^{t}\binom{n}{i}$, with $t = \left\lfloor\frac{d-1}{2}\right\rfloor$ where $\lfloor x \rfloor$ represents the largest integer less than or equal to x. For any error pattern e, of length n and weight w less than or equal to the correction capability t, we compute its binary syndrome S(e)=e. H^T and add it into X, in the corresponding row (of the same index) we store in Y the integer that represents the correctable error e associated with S.

3. SIMULATION RESULTS OF EL-BOOSTDEC AND DISCUSSION

In order to show the efficiency of the proposed decoder, we present in this section its error correction performances for some linear block codes of different lengths. The performances are represented in terms of BER versus signal to noise ratio (SNR) in the additive white gaussian noise channel (AWGN), and with binary phase shift keying (BPSK) modulation. Parameters of simulation are:

– The minimum number of residual bits in error is equal to 200.
– The minimum number of transmitted blocks is equal to 100000.
– The basic model used in the boosting process of EL-BoostDec is the decision tree model.
– The number of models used in the boosting process depends on the studied code and will be given for each code.
– The depth of a tree is the maximum distance between the root and any leaf. The max depth value of trees used in the EL-BoostDec decoder will be given for each code.
– Learning rate = 0.2.

3.1. EL-BoostDec performances for some optimal codes of Joundan

The proposed EL-BoostDec decoder is used to decode some optimal codes constructed by genetic algorithms [16] with the parameters listed in the Table 2. The corresponding results in terms of BER versus SNR are given in the Figure 1. From Figure 1, it can be deduced that the EL-BoostDec decoder used to decode the J(52, 39, 6) code reaches a BER equal to 10^{-5} at the SNR 7.6 dB. When it is used to decode the J(28, 21, 4) code, it achieves the same BER at SNR 8.5 dB.

Table 2. Parameters of EL-BoostDec for some Joundan codes

Joundan code	The number of models used in the boosting	Max depth value
J(26, 13, 7)	2	13
J(28, 21, 4)	1	10
J(48, 32, 6)	5	15
J(52, 39, 6)	5	15

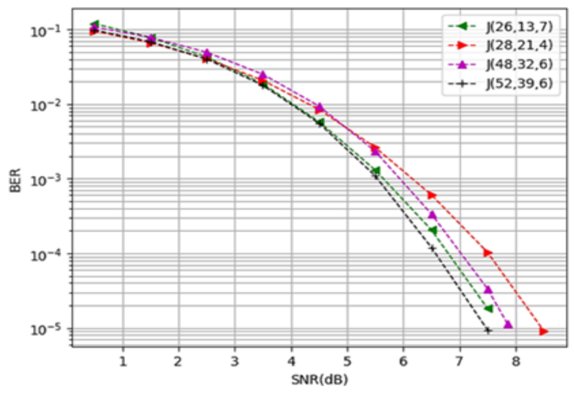

Figure 1. Performances of EL-BoostDec for some Joundan codes

3.2. EL-BoostDec performances for some BCH codes

The proposed EL-BoostDec decoder is used to decode some BCH codes using the following parameters given in the Table 3. The corresponding results in terms of BER versus SNR are given in the Figures 2 to 4. From Figure 2, we deduced that the EL-BoostDec decoder used to decode the BCH(15, 5, 7) code reaches a BER equal to 10^{-5} at the SNR 8.7 dB. We notice that the BCH(15, 7, 5) has relatively the same performances as BCH(15, 5, 7).

From Figure 3, we deduced that the EL-BoostDec decoder used to decode the BCH(31, 16, 7) code reaches a BER equal to 10^{-5} at the SNR 7.6 dB. When it is used to decode the BCH(31, 26, 3) code, it achieves the same BER at SNR 8.2 dB, i.e. a coding gain of 0.6 dB. From Figure 4, we deduced that the EL-BoostDec decoder used to decode the BCH(63, 51, 5) code reaches a BER equal to 10^{-5} at the SNR 7.4 dB, i.e. a coding gain of about 2.2 dB. When it is used to decode the BCH(63, 57, 3) code, it achieves the same BER at SNR 8 dB, i.e. a coding gain of about 1.6 dB.

Table 3. Parameters of EL-BoostDec for some BCH codes

BCH Code	The number of models used in the boosting	Max depth value
BCH(15,7,5)	2	8
BCH(15,5,7)	2	10
BCH(31,16,7)	2	15
BCH(31,21,5)	2	10
BCH(31,26,3)	1	5
BCH(63,51,5)	2	12
BCH(63,57,3)	2	6

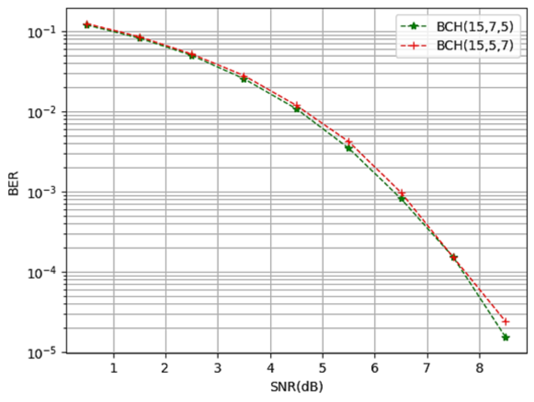

Figure 2. Performances of EL-BoostDec for some BCH codes of length 15

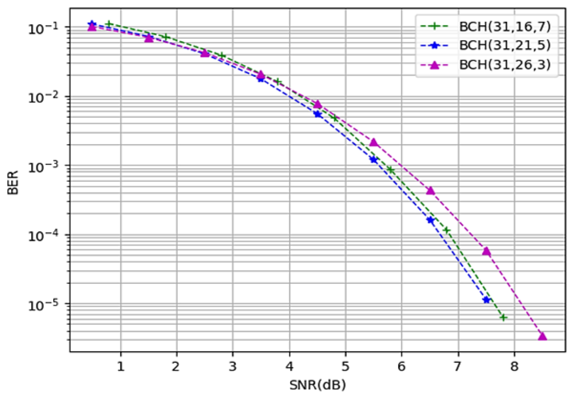

Figure 3. Performances of EL-BoostDec for some BCH codes of length 31

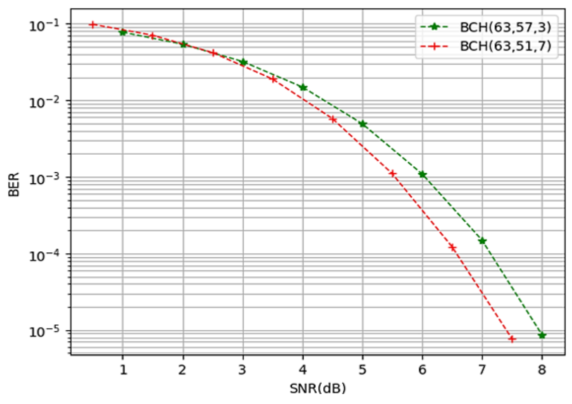

Figure 4. Performances of EL-BoostDec for some BCH codes of length 63

3.3. EL-BoostDec performances for some quadratic residue codes

The proposed EL-BoostDec decoder is used to decode some QR codes and the results are shown in Figure 5. It can be deduced that the EL-BoostDec decoder used to decode the QR(23, 12, 7) code (the number of models used in the boosting=1 and max depth value=8) reaches a BER equal to 10^{-5} at the SNR 7.5 dB, i.e. a coding gain of about 2.1 dB. When it is used to decode the QR(17, 9, 5) code, it achieved the same BER at SNR 8 dB, which corresponds to a coding gain of about 1.6 dB.

3.4. EL-BoostDec performances for the Golay code

Deploying the proposed EL-BoostDec decoder for the decoding of QR(24, 12, 8) code, we configured the boosting process with three models and a maximum depth value of 13. The outcomes, illustrated in Figure 6, distinctly demonstrate that the EL-BoostDec decoder, in its application to the QR(24, 12, 8) code, achieves a BER of 10^{-5} at a SNR of approximately 7.4 dB. This remarkable performance improvement corresponds to a coding gain of approximately 2.2 dB, emphasizing the efficacy the EL-BoostDec algorithm in enhancing the error correction capabilities of the QR code.

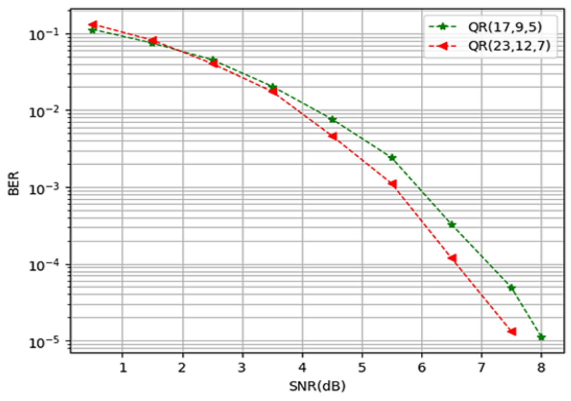

Figure 5. Performances of EL-BoostDec for some QR codes of length 17 and 23

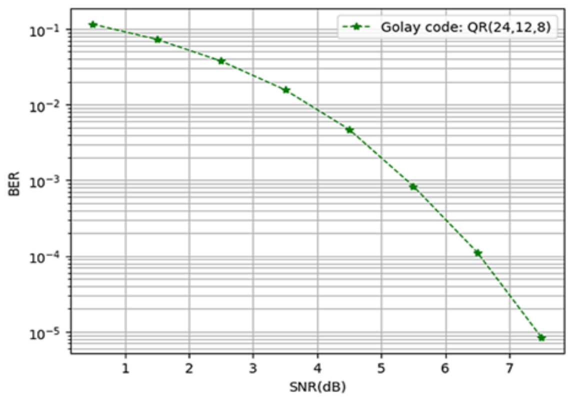

Figure 6. Performances of EL-BoostDec for QR (24, 12, 8) code

3.5. Study of the impact of EL-BoostDec parameters on their performances

Applying the proposed EL-BoostDec decoder to decode the QR(24, 12, 8) code, we configured the boosting process with three models and set the maximum depth value to 13. The results, as depicted in Figure 6, clearly illustrate that the EL-BoostDec decoder achieves a BER of 10^{-5} at a SNR of around 7.4 dB. This outcome indicates a significant coding gain of approximately 2.2 dB, showcasing the decoder's effectiveness in enhancing the error correction capabilities of the QR(24, 12, 8) code.

3.5.1. Impact of the number of models (decision tree)

In order to show the impact of the number of models (decision tree) on EL-BoostDec performances we have decoded the BCH(31, 16, 7) code. The value of max depth was fixed on 14 and the learning rate was set at 0.2, while the number of models used in the boosting process of EL-BoostDec was varied. The Figure 7 shows the impact of the number of models, it confirms that the increase of the number of models improve considerably the performances. From 1 model to 3 models a gain of about 2.8 dB is benefited at BER=10^{-5}. But the use of 9 models doesn't give important improvements comparing to the 3 models version; however, the increase of the number of used models increases also the temporal complexity of the EL-BoostDec decoder, this means that this parameter should be adapted for each code to find the minimum number of models that yield to good performances.

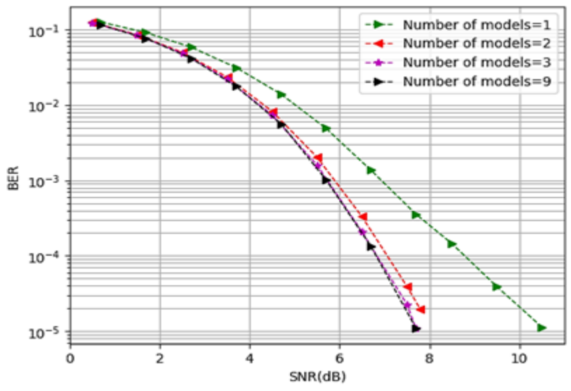

Figure 7. Performances of the number of models on EL-BoostDec performances

3.5.2. Impact of the max depth value

In order to show the impact of the number of models (decision tree) on EL-BoostDec performances we have decoded the BCH(31, 16, 7) code by fixing the learning rate at 0.2 and the number of models at 2 and varying the value of max depth between 5 and 15. The Figure 8 shows the impact of the max depth value, it confirms that the increase of the max depth value improves considerably the performances. From the max depth value equal to 5 to the value 15 about 4 dB is benefited at only BER=10^{-3}. On the values of BER=10^{-5} the gain is very large. We have also studied some values more than 15 and they don't give important improvements comparing to the value 15; however, the increase of this max depth value increase also the temporal complexity of the EL-BoostDec decoder, this means that this parameter should be adapted for each code to find the minimum max depth value that yield to good performances.

3.5.3. Impact of the learning rate parameter

To assess the impact of the learning rate on EL-BoostDec's performance, we conducted decoding experiments on the BCH(31, 16, 7) code. Keeping the max depth value fixed at 15 and the number of models at 2, we systematically varied the learning rate within the range of 0.1 to 0.3. Surprisingly, the results, illustrated in Figure 9, indicate that the performance of the EL-BoostDec decoder remains largely unaffected by changes in the learning rate, at least within the specified range, for this particular code. This suggests that, for the given configuration and code, the learning rate may not be a critical factor influencing the decoder's efficacy.

3.6. Comparison between EL-BoostDec and some competitors

3.6.1. Comparison with logistic regression decoder [8]

The Table 4 gives a comparison of scores between EL-BoostDec and logistic regression decoder (LRDec) [8] accuracy in the training phase. It indicates the score for some codes in the training process. Knowing that, the training set contains all possible syndromes, therefore the performances in terms of BER correction of EL-BoostDec passes considerably those of LRDec decoder for the BCH(63, 51, 5) code. For the codes BCH(15, 5, 7) and BCH(31, 16, 7) they have the same performances. This comparison demonstrates clearly that when the code length increases, the EL-BoostDec remains powerful but the efficiency of LRDec decrease.

3.6.2. Comparison of EL-BoostDec, HSDec, ARDEcGA, and BERT decoders [10], [12], [25]

Figure 10 provides a comprehensive performance comparison of decoding algorithms, including EL-BoostDec, hash and syndrome decoding (HSDec) [10], ARDEcGA [12], and bit error rate test (BERT) decoder [25], specifically applied to the BCH(15, 7, 5) code. The analysis of this comparison reveals that EL-BoostDec exhibits comparable performance to HSDec and ARDEcGA, and notably, it surpasses the BERT decoder for this particular code. This insight underscores the competitive and effective nature of the EL-BoostDec algorithm in decoding BCH(15, 7, 5) codes, showcasing its potential as a robust error-correction tool.

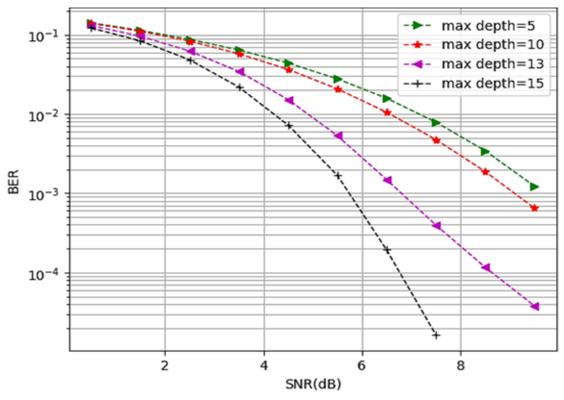

Figure 8. The impact of the max depth value on EL-BoostDec performances

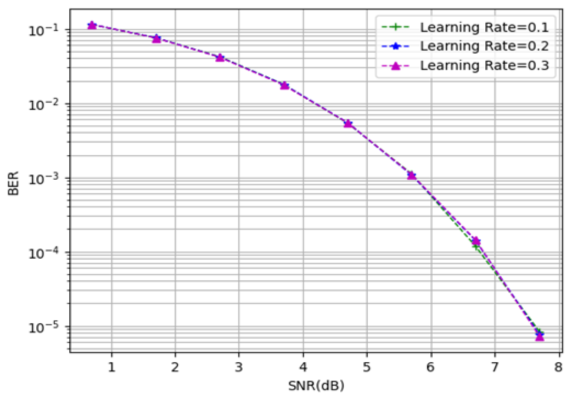

Figure 9. The impact of the learning rate parameter on EL-BoostDec performances

Table 4. Comparison of accuracy between EL-BoostDec and LRDec decoders

Linear codes	Score model	
	LRDec (%)	EL-BoostDec (%)
BCH(15, 5, 7)	100	100
BCH(31, 16, 7)	100	100
BCH(63, 51, 5)	96	100

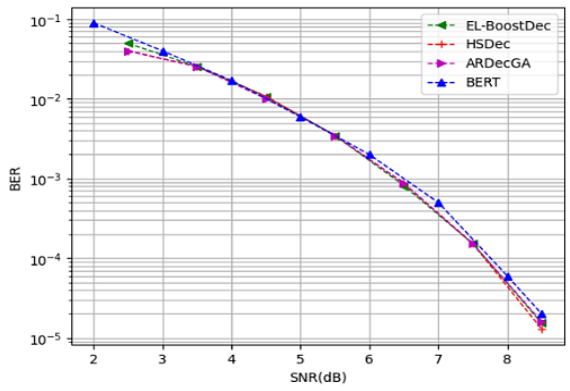

Figure 10. Performances of EL-BoostDec, HSDec, ARDEcGA, and BERT decoder for BCH(15, 8, 5) code

3.6.4. Comparison of EL-BoostDec and the decoder of Sahu *et al*. [14]

Sahu *et al.* [14] introduces an algebraic decoder designed for the QR(17, 9, 5) code. A thorough comparison between this algebraic decoder and EL-BoostDec, specifically applied to the QR(17, 9, 5) code, reveals a noteworthy finding. Both decoders demonstrate equivalent performance, particularly in their ability to successfully decode errors of weights less than or equal to 2. This parity in performance suggests that, for the QR(17, 9, 5) code, EL-BoostDec stands on equal footing with the algebraic decoder introduced in [14], emphasizing its competence in error correction.

3.6.5. Comparison of EL-BoostDec and the decoder of Chien [6]

Chien [6] present a decoder based on specific mapping. The comparison of the power of this decoder with that of EL-BoostDec for the QR(23, 12, 7) code demonstrate that they have the same performances in terms of their capabilities to decode all errors of weights less than or equal to 3. Here, the main advantage of EL-BoostDec is that it stores only the intelligent machine learning model instead storing the syndromes and their corresponding error patterns in [6].

3.6.6. Comparison of EL-BoostDec and the decoder of Boualame *et al*. [13]

Boualame *et al.* [13] introduce an algebraic decoder specifically designed for the QR(17, 9, 5) code. This decoder's efficacy is compared with EL-BoostDec, revealing comparable performance levels for decoding errors of weights up to 2. This comparison demonstrate that they have the same performances for this code.

3.6.7. Comparison of EL-BoostDec and the simulated annealing decoder of Aylaj and Belkasmi [17]

Aylaj and Belkasmi [17] introduces a novel variant of the SA method for error correction. To evaluate its efficacy, a performance comparison was conducted between EL-BoostDec and this SA decoder, specifically applied to the BCH(31, 16, 7) and BCH(15, 7, 5) codes. Remarkably, the comparison reveals that both decoders yield identical results for these codes, suggesting a comparable level of performance in error correction. This finding underscores the robustness of EL-BoostDec, showcasing its effectiveness alongside a state-of-the-art SA decoder in the context of BCH codes.

4. CONCLUSION AND PERSPECTIVES

Among the most powerful machine learning techniques, the aggregation methods (adaptive methods or averaging) of models have given excellent results in many research areas. It is this excellence that led us to think of their application on the decoding problem. In this sense, we have successfully designed, tested and validated successfully our EL-BoostDec decoder which is based on the calculation of the syndrome and on the use of ensemble learning techniques to find the error. The results of the proposed EL-BoostDec are very encouraging in terms of the BER that it offers, with ability to guarantee 100% correction of errors with weights less than or equal to the correction capability of the studied codes. In perspectives, we plan to apply EL-BoostDec on other codes and to make another version by using bagging models.

REFERENCES

[1] K. Knight, “Decoding complexity in word-replacement translation models,” *Computational linguistics*, vol. 25, no. 4, pp. 607–615, 1999.

[2] E. R. Berlekamp, R. J. McEliece, and H. C. A. van Tilborg, “On the inherent intractability of certain coding problems,” *IEEE Transactions on Information Theory*, vol. 24, no. 3, pp. 384–386, 1978, doi: 10.1109/TIT.1978.1055873.

[3] I. S. Reed, X. Yin, T. K. Truong, and J. K. Holmes, “Decoding the (24,12,8) Golay code,” *IEE Proceedings E: Computers and Digital Techniques*, vol. 137, no. 3, pp. 202–206, 1990, doi: 10.1049/ip-e.1990.0025.

[4] I. S. Reed and T. K. Truong, “Algebraic decoding of the (32,16,8) quadratic residue code,” *IEEE Transactions on Information Theory*, vol. 36, no. 4, pp. 876–880, 1990, doi: 10.1109/18.53750.

[5] T. K. Truong, X. Chen, and X. Yin, “The algebraic decoding of the (41, 21, 9) quadratic residue code,” *IEEE Transactions on Information Theory*, vol. 38, no. 3, pp. 974–986, 1992, doi: 10.1109/18.135639.

[6] C.-H. Chien, “Developing efficient algorithms of decoding the systematic quadratic residue code with lookup tables,” *International Journal of Operations Research*, vol. 13, no. 4, pp. 165–174, 2016.

[7] E. Nachmani, Y. Be’Ery, and D. Burshtein, “Learning to decode linear codes using deep learning,” in *54th Annual Allerton Conference on Communication, Control, and Computing, Allerton 2016*, IEEE, Sep. 2017, pp. 341–346. doi: 10.1109/ALLERTON.2016.7852251.

[8] C. I. Imrane, N. Said, B. El Mehdi, E. K. A. Seddiq, and M. Abdelaziz, “Machine learning for decoding linear block codes: case of multi-class logistic regression model,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 24, no. 1, pp. 538–547, 2021, doi: 10.11591/ijeecs.v24.i1.pp538-547.

[9] E. Nachmani, E. Marciano, D. Burshtein, and Y. Be’ery, “RNN decoding of linear block codes,” *arXiv-Computer Science*, pp. 1-7, 2017.

[10] 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 *Lecture Notes in Real-Time Intelligent Systems*, Cham: Springer, 2019, pp. 448–459. doi: 10.1007/978-3-319-91337-7_40.

[11] S. -I. Chu, S. -A. Ke, S. -J. Liu, and Y. -W. Lin, “An efficient hard-detection GRAND decoder for systematic linear block codes,” in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 31, no. 11, pp. 1852-1864, Nov. 2023, doi: 10.1109/TVLSI.2023.3300568

[12] S. Nouh, A. El Khatabi, and M. Belkasmi, “Majority voting procedure allowing soft decision decoding of linear block codes on binary channels,” *International Journal of Communications, Network and System Sciences*, vol. 05, no. 09, pp. 557–568, 2012, doi: 10.4236/ijcns.2012.59066.

[13] H. Boualame, I. Chana, and M. Belkasmi, “New efficient decoding algorithm of the (17, 9, 5) quadratic residue code,” in *Proceedings - 2018 International Conference on Advanced Communication Technologies and Networking, CommNet 2018*, IEEE, Apr. 2018, pp. 1–6. doi: 10.1109/COMMNET.2018.8360258.

[14] R. Sahu, B. P. Tripathi, and S. K. Bhatt, “New algebraic decoding of (17, 9, 5) quadratic residue code by using inverse free berlekamp-massey algorithm (IFBM),” *International Journal of Computational Intelligence Research (IJCIR)*, vol. 13, no. 8, pp. 2015–2027, 2017.

[15] L. Niharmine, H. Bouzkraoui, A. Azouaoui, and Y. Hadi, “Simulated annealing decoder for linear block codes,” *Journal of Computer Science*, vol. 14, no. 8, pp. 1174–1189, Aug. 2018, doi: 10.3844/jcssp.2018.1174.1189.

[16] I. A. Joundan, S. Nouh, and A. Namir, “Design of good linear codes for a decoder based on majority voting procedure,” in *2016 International Conference on Advanced Communication Systems and Information Security, ACOSIS 2016 - Proceedings*, IEEE, 2017. doi: 10.1109/ACOSIS.2016.7843918.

[17] B. Aylaj and M. Belkasmi, “Simulated annealing decoding of linear block codes,” in *Proceedings of the Mediterranean Conference on Information & Communication Technologies 2015: MedCT 2015 Volume 1*, Springer International Publishing, 2016, pp. 175–183. doi: 10.1007/978-3-319-30301-7_19.

[18] S. El K. Alaoui, Z. Chiba, H. Faham, M. El Assad, and S. Nouh, “Efficiency of two decoders based on hash techniques and syndrome calculation over a Rayleigh channel,” *International Journal of Electrical and Computer Engineering*, vol. 13, no. 2, pp. 1880–1890, Apr. 2023, doi: 10.11591/ijece.v13i2.pp1880-1890.

[19] X. Ruan, “Deep learning algorithms for BCH decoding in satellite communication,” *Highlights in Science, Engineering and Technology*, vol. 38, pp. 1104–1115, 2023, doi: 10.54097/hset.v38i.6012.

[20] X. Chen and M. Ye, “Neural decoders with permutation invariant structure,” *Journal of the Franklin Institute*, vol. 360, no. 8, pp. 5481–5503, 2023, doi: 10.1016/j.jfranklin.2023.03.024.

[21] D. Khebbou, I. Chana, and H. Ben-Azza, “Decoding of the extended Golay code by the simplified successive-cancellation list decoder adapted to multi-kernel polar codes,” *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 21, no. 3, pp. 477–485, 2023, doi: 10.12928/TELKOMNIKA.v21i3.23360.

[22] H. Boualame, M. Belkasmi, and I. Chana, “Efficient decoding algorithm for binary quadratic residue codes using reduced permutation sets,” *Journal of Computer Science*, vol. 19, no. 4, pp. 526–539, Apr. 2023, doi: 10.3844/jcssp.2023.526.539.

[23] G. Kunapuli, *Ensemble methods for machine learning*, Shelter Island, New York: Simon and Schuster, 2023.

[24] K. W. Walker, “Exploring adaptive boosting (AdaBoost) as a platform for the predictive modeling of tangible collection usage”, *The Journal of Academic Librarianship*, vol. 47, no. 6, 2021, doi: 10.1016/j.acalib.2021.102450.

[25] M. Elghayyaty, A. Hadjoudja, O. Mouhib, A. El Habti, and M. Chakir, “Performance Study of BCH error correcting codes using the bit error rate term BER,” *International Journal of Engineering Research and Applications*, vol. 7, no. 2, pp. 52-54, 2017, doi: 10.9790/9622-0702025254.

BIOGRAPHIES OF AUTHORS

Mohammed El Assad is Ph.D. Student at Faculty of sciences Ben M’Sik (FSBM), Hassan II University, Casablanca, Morocco. He had master thesis in computer sciences at FSBM in 2021. His current research interest is networks and systems, telecommunications, information, coding theory, machine learning, and deep learning. He can be contacted at email: mohammed.elassad-etu@etu.univh2c.ma.

Said Nouh holds a Ph.D. in Computer Sciences at National School of Computer Science and Systems Analysis (ENSIAS), Rabat, Morocco in 2014. He is currently professor (higher degree research (HDR)) at Faculty of Sciences Ben M'Sick, Hassan II University, Casablanca, Morocco. His current research interests are artificial intelligence, machine learning, deep learning, telecommunications, information, and coding theory. He can be contacted at email: said.nouh@univh2m.ma.

Imrane Chemseddine Idrissi is a Ph.D. student at Faculty of Sciences Ben M'Sik (FSBM), Hassan II University, Casablanca, Morocco. He received a master's thesis in data science and big data at ENSIAS Mohammed V university in 2019. His current research interests include networks and systems, telecommunications, information, coding theory, machine learning, and deep learning. He can be contacted at email: imrane.chemseddine-etu@etu.univh2c.ma or imran.chems@gmail.com.

Seddiq El Kasmi Alaoui is an assistant professor in the Department of Mathematics and Computer Science at Faculty of Sciences, Hassan II University of Casablanca, Morocco. He received his Ph.D. degree in Computer Scinece from the same university, a Master in networks and systems from Hassan I University. His current research interest is information and coding theory, machine learning, and deep learning. He can be contacted at email: sadikkasmi@gmail.com.

Bouchaib Aylaj received his joint Ph.D. degree in computer science from faculty of sciences, Chouaib Doukkali University, and ENSIAS, Med.V University, Morocco in 2016. He is currently an assistant professor in the CRMEF-Rabat Morroco. His research interests include artificial intelegence, error control coding for digial communications, digital signal processing, and embedded systems. He can be contacted at email: bouchaib_aylaj@yahoo.fr.

Mohamed Azzouazi is a professor at Faculty of Sciences Ben M'Sik, Hassan II University of Casablanca, Morocco. He obtained his doctoral thesis in automatic processing of natural languages at Mohammadia School of Engineers, Rabat, Morocco in 1997. His current research interests constraint of satisfaction problem, big data, predictive analysis, deep learning, and other fields of computer science. He can be contacted at email: MohamedAzzouazi@yahoo.fr.