Machine Learning for Channel Coding: A Paradigm Shift from FEC Codes

Kayode A. Olaniyi[1], Reolyn Heymann[1], and Theo G. Swart[1, 2, *]

[1] Department of Electrical and Electronic Engineering Science, University of Johannesburg, South Africa
[2] Center for Telecommunications, University of Johannesburg, South Africa
Email: 217097911@student.uj.ac.za (K.A.O.); reolyn.heymann@gmail.com (R.H.); tgswart@uj.ac.za (T.G.S.)
*Corresponding author

***Abstract*—The design of optimal channel codes with computationally efficient Forward Error Correction (FEC) codes remains an open research problem. In this paper, we explore optimal channel codes with computationally efficient FEC codes, focusing on turbo and Low-Density Parity-Check (LDPC) codes as near-capacity approaching solutions. We highlight the significance of accurate channel estimation in reliable communication technology design. We further note that the stringent requirements of contemporary communication systems have pushed conventional FEC codes to their limits. To address this, we advocate for a paradigm shift towards emerging Machine Learning (ML) applications in communication. Our review highlights ML's potential to solve current channel coding and estimation challenges by replacing traditional communication algorithms with adaptable deep neural network architectures. This approach provides competitive performance, flexibility, reduced complexity and latency, heralding the era of ML-based communication applications as the future of end-to-end efficient communication systems.**

***Keywords*—Turbo codes, LDPC codes, autoencoder, interleaver, encoder**

I. Introduction

Communication systems are crucial for enabling the exchange of information across vast distances in modern society. Channel coding is a fundamental technique used in communication systems to ensure reliable and secure data transmission by addition of redundant bits to the original message for error correction. Traditional channel coding schemes, such turbo codes and Low-Density Parity-Check (LDPC) codes, have been widely used and optimized to achieve near-capacity performance [1]. However, these conventional coding schemes are based on mathematical models and heuristics, which may not fully exploit the potential of modern communication systems, especially in non-linear systems.

With the emergence of Machine Learning (ML) techniques, there has been growing interest in leveraging ML for communication applications, including end-to-end channel coding and estimation [2]. ML-based approaches have the potential to adaptively learn and optimize coding schemes based on data-driven insights, leading to improved performance and adaptability to changing channel conditions.

A reliable communication system needs to transfer data accurately, with high throughput and low latency across a transmission channel [3]. However, communication channels have limitations, such as hostile environments that can result in unreliable signal propagation [1]. Thus, to design communication technologies that are reliable and energy-efficient, accurate knowledge of the communication channel is essential. Unfortunately, and in reality, the complexity of communication media, especially in terms of noise or interference, is not fully understood or modeled. Moreover, the demands of modern communication systems, such as 5G wireless communications, have further stretched the capabilities of conventional communication schemes. Developing strategies and efficient algorithms to mitigate these limitations and achieve reliable communication systems remains an open research problem.

The current requirements of communication systems call for near-capacity channel codes that are highly optimized and can combat noise in the communication medium [2, 4]. Fig. 1 depicts a classification diagram of the available error control schemes.

Forward Error Correction (FEC) is an error control scheme that aims to efficiently transmit data signals over noisy channels while enabling error-free decoding [2]. Fig. 2 illustrates a typical FEC model. In digital transmission, FEC coding schemes involve adding carefully designed redundancy to the original signal k to create a codeword n, which aids the receiver decoder in detecting and even correcting transmission errors [4, 5], FEC codes are commonly classified as block (algebraic) codes and convolutional codes, with varying complexity and performance. The authors in [6] emphasized that efficient encoding schemes for well-constructed input message patterns and appropriate decoding algorithms are critical design considerations for all FEC coding schemes.

Manuscript received August 8, 2023; revised August 30, 2023; accepted November 1, 2023; published February 25, 2024.

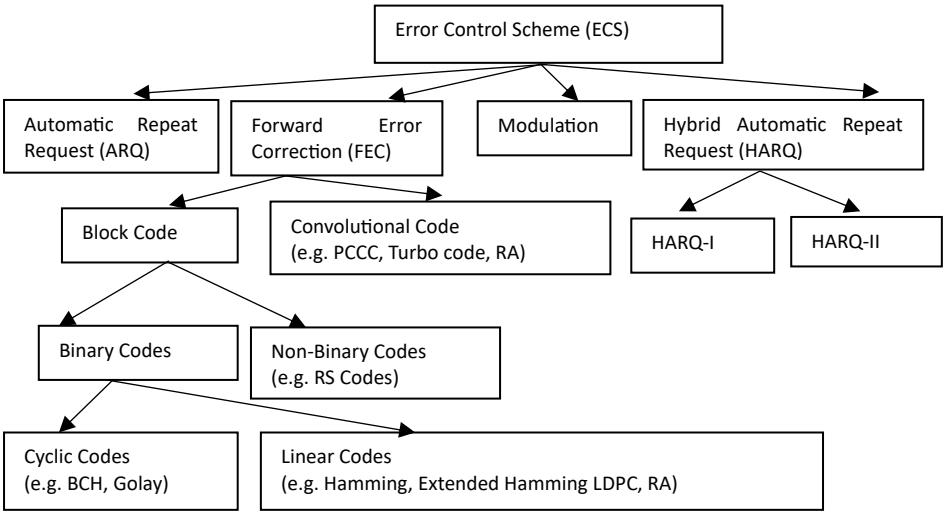

Fig. 1. Error control scheme (ECS) classification diagram.

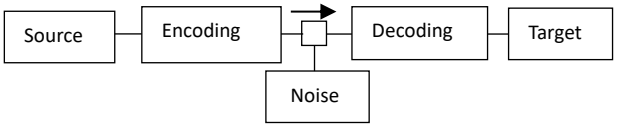

Fig. 2. FEC communication model diagram.

While the encoder aims to find an efficient codeword representation for the message to be transmitted over a noisy channel, the decoder aims to identify the most probable codeword sent. At a low signal-to-noise ratio (SNR) where there is considerable noise interference, a low coding rate ($R = k/n$) is required. However, this necessitates adding significant redundancy, increasing bandwidth and energy consumption. This limitation is a major factor that restricts the deployment of FEC codes in high-throughput applications.

FEC techniques are typically preferred when signal re-transmission is impractical or costly. Ideally, FEC codes should successfully detect and correct errors and enable transmission at coding rates approaching the Shannon capacity [2].

Numerous FEC coding strategies have been proposed to achieve near-optimal channel capacity while minimizing costs, encoding/decoding complexity, power consumption and transmitting errors [7]. Despite significant progress in the field, some current FEC strategies suffer from high encoding/decoding complexity, increased power consumption, and redundant bit overhead [7]. Additionally, the error-floor effect in the high SNR region poses a challenge, where the Bit Error Rate (BER) curve shows a significantly decreased slope compared to lower SNR regions. This paper aims to contribute potential solutions to these challenges.

The remainder of the paper is organized as follows. Section II explores the principles and characteristics of near-optimal capacity-approaching FEC channel codes, with turbo codes and LDPC codes being considered the two most powerful and commonly used FEC codes in modern communication systems [8]. This section provides specific turbo and LDPC codes performance metrics as examples of near capacity-approaching FEC codes, including communication channel throughput, BER, latency and complexity. Section III presents an overview of current ML approaches for achieving near-optimal channel coding schemes, which have the potential to overcome the limitations of canonical FEC channel coding techniques. Finally, Section IV concludes the paper with recommendations.

II. Near-Capacity Approaching Channel Coding

In this section, we conduct a review of code classes that are capable of approaching the Shannon limit closely. Specifically, we analyze and compare turbo codes and low-Density Parity-Check (LDPC) codes, both of which involve iterative decoding evaluated based on various metrics, including encoding and decoder schemes, system complexity, bandwidth, throughput, BER or SNR, memory usage and latency.

A. Turbo Codes

Turbo codes are parallel concatenated convolutional codes that are widely used in the field of communication [1, 2, 9]. They were first introduced in 1993 by Berrou *et al*. [3] and Arif *et al.* [10]. Turbo codes achieved impressive performance, coming within 0.7 dB of Shannon's channel capacity limit at a BER of 10^{-5} on an Additive White Gaussian Noise (AWGN) channel [5] Turbo codes typically involve using two or more parallel concatenated convolutional encoders with pseudorandom interleavers and employ a converging iterative decoding procedure that feeds the outputs of one decoder to the inputs of another using a soft-decision algorithm [1, 5–7] Hard-output decoders are generally not suitable as they can degrade system performance. Instead, Soft-Input/Soft-Output (SISO) decoders are commonly adopted for Robust decoding of turbo codes. However, the number of iterations required for convergence can result in latency drawbacks, which impose limits on the block length n.

The performance of turbo codes in terms of BER is not

only influenced by designing more powerful encoders/decoders or increasing the encoder/decoder dimension but also by the design of the constituent interleavers [7, 8] . Two important factors that are commonly considered significant in evaluating turbo code performance are the convergence behavior of iterative decoding in the low SNR region, known as the "waterfall" region, and the "error-floor" effect in the high SNR region [11, 12]. Conventional turbo codes that use 8-state constituent encoders can perform effectively at low SNR but often suffer from a flattening around a Frame Error Rate (FER) of 10^{-5} due to a large number of low-weight codewords, which corresponds to a poor minimum distance, $d_{\min}$ [8]. It has been shown that modifying interleaver parameters and puncturing patterns, such as encoding information bits of various lengths with different coding rates, can improve the BER performance [8]. According to the authors in [11] the "error-floor" challenge is becoming less problematic due to good permutation matrices.

B. Turbo Encoding

A pioneering work [13] proposed a convolutional code with near Shannon capacity, utilizing a parallel concatenation of two Recursive Systematic Convolutional (RSC) codes. This approach was shown to outperform the best non-Systematic Convolutional (NSC) code at any SNR for high code rates, as demonstrated in [14]. The authors compared BER performance and concluded that while NSC was superior to Systematic Convolutional (SC) code at high SNR and vice versa at low SNR, the RSC offered good performance at both low and high SNR, as shown in [14], Fig. 3. However, one drawback of RSC codes is the lack of long-range memory [15]. To address this limitation, the authors in [13] introduced long-range memory by concatenating the first encoder's output with the second encoder's interleaved output.

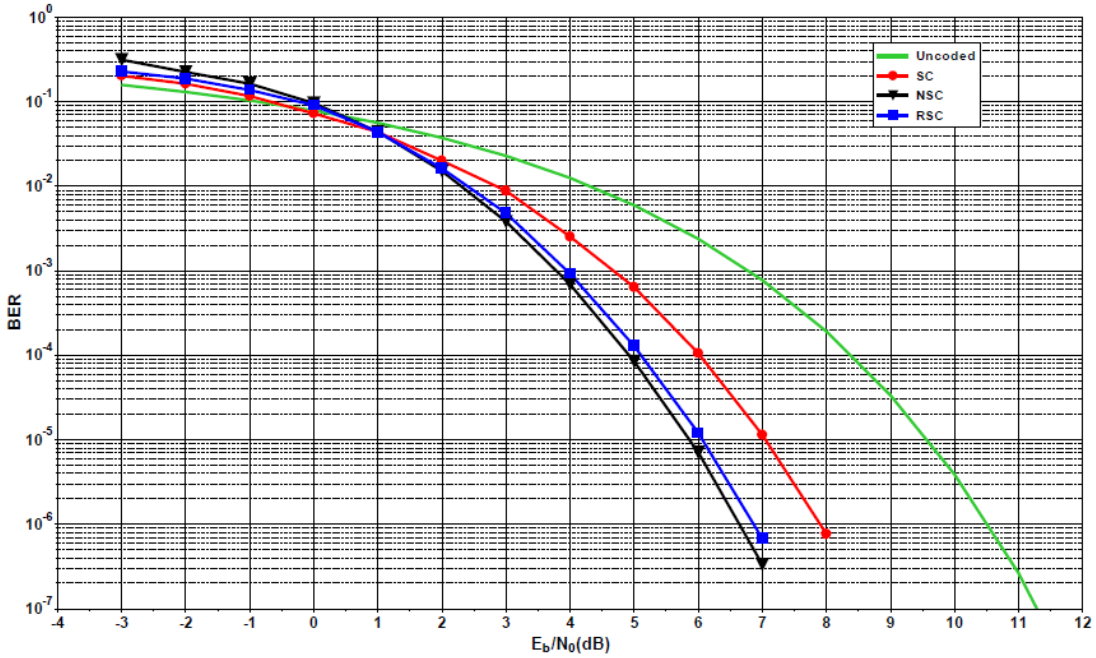

Fig. 3. Simulated performance of the memory-2 RSC code with NSC and SC codes [14].

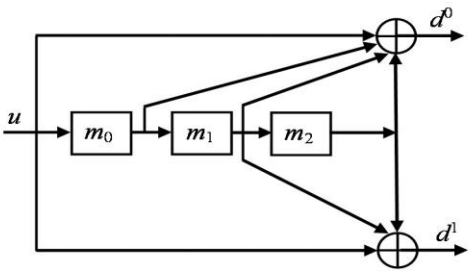

Fig. 4. R = 1/2 non-systematic feed-forward convolutional encoder with memory m = 3.

The relationship between the constraint length k of a convolutional code and the maximum number of memory stages m in the encoder is defined as $k = m + 1$ [16]. To minimize decoding computational complexity, typically, m is chosen to be between 3 and 5. Fig. 4 illustrates a ½ rate NSC encoder with $m = 3$.

This NSC encoder has two generator functions denoted, g^0 and g^1 and can be represented algebraically:

$$g^1 = 1 + x + x^2 + x^3$$

$$g^0 = 1 + x^2 + x^3$$

The parity bits d^0 and d^1 are generated as the modulo-2 addition of the message bit u and the respective generator functions:

$$d^0 = u \oplus g^0 \quad (1)$$

$$d^1 = u \oplus g^1 \quad (2)$$

Fig. 5 depicts the block diagram of the conventional turbo encoder [3].

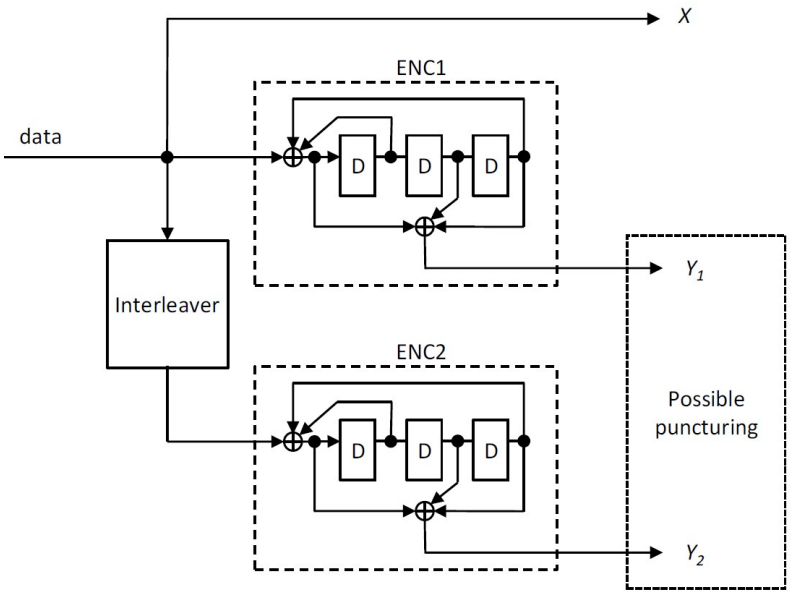

Fig. 5. Classical turbo encoder structure.

In this architecture, the first RSC encoder (ENC1) takes a binary information bit sequence of length k as input and generates an encoded stream codeword y^1. Simultaneously, this information bit sequence is reordered by an interleaver and then encoded by the second RSC encoder (ENC2) to produce another codeword y^2, which behaves like a long random code [17]. The information sequence X, along with the two parity sequences y^1 and y^2, are transmitted with a combined code rate R of 1/3. In cases where different code rates are desired, puncturing elements can be included to puncture and multiplex the encoded bit sequences before they are modulated and transmitted over the physical channel. As stated in [14] the global coding rate for a parallel concatenation of two elementary codes, C_1 and C_2, with coding rates R_1 and R_2, can be represented by the expression:

$$R_p = \frac{R_1 R_2}{R_1 + R_2} = \frac{R_1 R_2}{1 - (1 - R_1)(1 - R_2)}. \quad (3)$$

As per Shannon's pioneering work [3, 7] random-like codes are essential for approaching capacity in communication systems. In [2], the authors proposed the use of the random-like (R-L) criterion as a basis for designing turbo codes. The R-L algorithm distinguishes between strongly and weakly random-like codes by measuring the closeness of their weight distribution to the average weight distribution obtained from random coding. The authors concluded that incorporating these codes as components in turbo code schemes can enhance the low-weight tail of the distribution and allow for adjustments to the BER according to specific specifications.

C. Interleaver

Extensive research has been conducted on interleaver design to achieve effective randomization and substantially enhance the code distance properties [4, 12, 18, 19]. The design strategy of interleavers plays a crucial role in determining the achievable minimum Hamming distance of the code, as well as the suitability for iterative decoding based on the IDS criterion [10], Interleavers are utilized in a specific pattern to randomize the location of errors and reduce the correlation between parity bits corresponding to the original and interleaved data frames, as low-weight generated codes often result in poor error performance [7, 10, 20] To avoid the costly use of storage elements or look-up tables associated with randomly selected permutations, interleaver designs employing algebraic permutation methods are generally preferred, as mentioned in [18]. However, it is worth noting that most of the research on interleaver design is based on the *S*-random algorithm [10, 21]

In [22], a method for designing efficient puncture-constrained interleavers was introduced, where Garzón-Bohórquez *et al*. claimed that applying interleaving with a periodic cross-connection pattern resembling a photograph not only improves the error-correction capability of the code but also significantly reduces the search space for different interleaver parameters. On the other hand, Onurcan *et al*. [23] proposed the concept of window band structured interleavers for turbo codes, where the generated symbols are limited within the window structure to improve the decoding latency characteristics. This proposed scheme allows for flexible trade-offs between latency and error correction performance.

D. Turbo Decoder

The utilization of intelligent encodings and reordering strategies discussed earlier leads to the generation of powerful codes that require complex decoding operations over multiple iterative soft-decision cycles [4, 5]. Turbo decoders, due to their iterative probabilistic nature, generally exhibit higher complexity compared to encoders.

Some of the challenges in training decoders are developing models that can successfully capture the underlying patterns in the data and transfer these continuous representations to discrete outputs, interference

latency, and experimenting with different model architectures and hyperparameters to achieve the desired performance. Fig. 6 illustrates a block diagram of a turbo decoder [4].

Turbo codes employ iterative decoding between SISO cascaded decoders, which extract systematic and recursive bits from the received information bit sequence and generate probabilities for each received bit being either 0 or 1. The maximum *a-posteriori* probability (MAP) and soft output Viterbi algorithm (SOVA) are two commonly used optimal decoding algorithms in turbo decoding. MAP involves *a-posteriori* probabilities, while SOVA is based on maximum-likelihood [5, 24, 25]. Turbo decoders typically operate in the log domain to reduce implementation complexity and time delay [8, 24, 26]. The decoders exchange log likelihood ratios (LLRs) for each binary input bit d^k, as expressed in [4, 27].

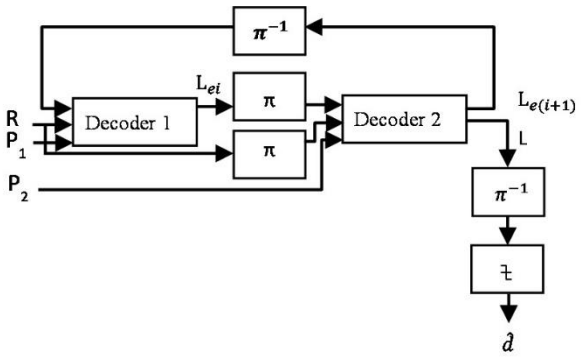

Fig. 6. Turbo decoder structure [4].

$$\ln\frac{\text{Prob}\{d^k=0|y,C\}}{\text{Prob}\{d^k=1|y,C\}}=L_n^k+\sum_i L_{ei}^k. \quad (4)$$

where C represents the code structure, y denotes the decoder inputs and i is the number of decoding iterations. The L_n^k and the L_{ei}^k are the intrinsic and extrinsic information decoder contributions during the decoding operations.

To obtain a block of k extrinsic LLRs, the addition and Jacobian logarithm operations (max* operation) used for combining two LLRs x and y is expressed in [1]:

$$\max^*(x.y)=\max(x,y)+\ln\left(1+e^{-|y-x|}\right). \quad (5)$$

In the max-log-MAP algorithm, the max* operations logarithm is approximated, as reported in [28], by

$$\max^*(x.y)\approx\max(x,y). \quad (6)$$

Once a certain number of SISO decoder operations, also known as half-iterations, have been completed, the turbo decoder produces estimates for the information bits by analyzing the sign of the intrinsic LLRs [29]

The comparative analysis of the decoding algorithms, as reported in [5], is summarized in Table I. Furthermore, a review of research on turbo codes is presented in Table II. The simulated BER performance of turbo codes, as reported by [8], is depicted in Fig. 7. Mensouri *et al.* presented the frame error rate results for a block size of k = 1024 using simple Max-Log-MAP algorithm under 6 decoding iterations.

Turbo codes have found successful applications in 3G and 4G systems due to their high reliability. However, the computational intensity of the decoding algorithm and the inherent high latency pose challenges to meeting the stringent low latency requirements of 5G systems and mobile devices [30, 31]. Reducing the latency of decoding has always been a challenging task, often involving trade-offs with performance. To achieve improved performance, a comprehensive understanding of the core components in terms of BER in the floor region, complexity and latency is necessary.

TABLE I: COMPARATIVE PERFORMANCE OF VARIOUS ALGORITHMS

Decoding Algorithm	SNR at 103	No. of Iterations
Log-MAP	3.8 dB	14
Max-Log-MAP	3.9dB	10
SOVA	5.0dB	10

E. Low-Density Parity-Check (LDPC) Codes

Panem *et al.* [34] analysed the high implementation complexity of the LDPC code that was originally developed by Gallager in 1962. However, LDPC codes were reinvented by MacKay and Neal [35], and today they are increasingly being considered and widely used in high-throughput emerging communication systems due to their performance that approaches the Shannon limit, especially with the belief propagation decoding algorithm [8, 36–38].

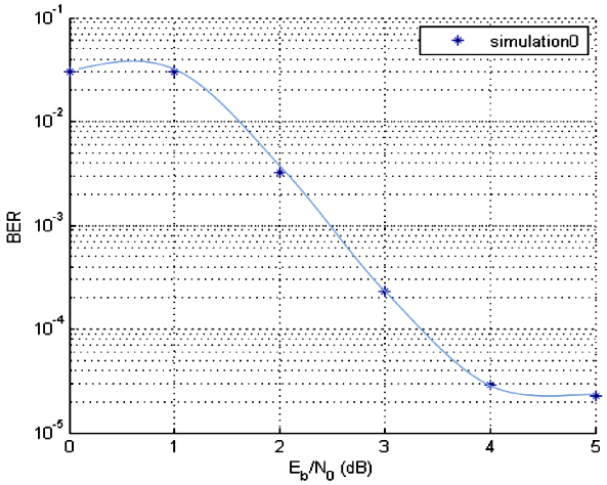

Fig. 7. Performance, in BER, of the turbo codes

Recent research has shown that LDPC codes can achieve comparable or even superior performance to turbo codes in certain cases, LDPC codes typically perform better than turbo codes at low SNR values (high noise levels). In such circumstances, LDPC codes are renowned for their exceptional error-correction abilities. The performance differences between turbo and LDPC codes

may not be very noticeable in moderate SNR levels. Both codes can deliver trustworthy communication, however, LDPC codes frequently hold a minor advantage. The performance gap between the two codes closes at a high SNR value (low noise level) showing that the turbo code can perform remarkably well. While turbo codes have a higher error floor, LDPC codes have a lower one. This means that when exceptionally low error rates are necessary, LDPC codes can offer more dependable communication. When iterative decoding is used, turbo code decoding may be more difficult than LDPC code decoding. In situations where computational complexity is a consideration, LDPC codes are frequently preferred [39, 41], thanks to their flexible and low-complexity encoder, as well as high-speed decoder [42–44].

TABLE II. A SUMMARY OF TURBO CODES

Ref.	Proposed Turbo	Methods (Approach)	Results	Problems/Limitation
Bohórquez *et al.* [22]	Interleaver	Photograph-based interleavers for punctured turbo codes	Improve code error correction and reduce the interleaver parameters search space.	Performance comparison of the proposed method with existing standards and techniques is not fully comprehensive.
Işcan and Xu [23]	Interleaver	Window-interleaved turbo codes	Latency reduction and makes parallel decoding with high throughput possible.	Reduction in error performance.
Kene and Kulat [5]	Decoder	Developed a max-log-MAP decoding algorithm (modified log-map decoding algorithm)	Hardware complexity reduction.	BER performance slightly degraded but better than SOVA.
Berrou *et al.* [13]	Turbo code	Recursive systematic convolutional (RSC) codes (Classical turbo code)	Performance better than the best non-systematic convolutional code.	Performances are at 0.7dB from Shannon's limit.
Arif *et al.* [10]	Interleaver	Deterministic interleaver	Provided the larger minimum distance for short-frame turbo codes by uniformly spreading the points in its smaller subsets, over the entire range of the information frame.	Does not provide a clear motivation for why reducing the correlation between parity bits is important for improving the decoding capability of the MAP decoder.
Ramasamy *et al.* [6]	Asymmetric turbo code	RSC encoder with heuristic generator polynomials in decimal and 3G interleaver	Performs well in both "waterfall" and "error floor" regions and with a coding gain of 0.5–0.8 dB.	Increased number of iterations.
J. Sodha [32]	Frame synchronization technique	Concept of probability surface metric within a modified MAP decoder	Reduction in the probability of false alarms.	Slight increase in the average number of information bits to be processed.
Tanriover *et al.* [33]	Turbo code	Multi-fold coding. Binomial weight distribution. Dividing a long information sequence into multiple sections of equal length and then permuting with separate interleavers.	Improved error performance.	Increased complexity and latency.
Tonnellier *et al.* [12]	Turbo codes	Turbo codes with CRC. Flip and check algorithm, most unreliable bits are identified based on their associated extrinsic information.	Lowering the error floor of turbo codes. Significant reduction in computation complexity.	The computational complexity grows exponentially with the chosen parameter q
Vucetic *et al.* and Panem *et al* [7, 34]	Interleaver	S-random interleaver. Increasing distances of low-weighted codewords.	A better performance relative to pseudorandom interleavers.	Lack of specificity on interleaver designs and simulation results.

LDPC codes are long linear binary block codes represented by a parity-check matrix (H) of size $M \times N$, $N > M$, where N columns represent the received encoded bits (codeword), and each of the M rows represents a parity-check equation [37]. The relationship between M, N, and K is given by $M = N - K$, where K represents the number of information bits. The matrix degree distribution refers to the number of non-zero entries in each row and column of the matrix. The row-weight and column-weight represent the number of ones in a row and column of the parity-check matrix, respectively [43]. In the Tanner graph representation, the rows and columns of the parity-check matrix correspond to the check and variable nodes, respectively. The distribution of variable nodes in polynomial representation on the Tanner graph can be expressed as:

$$\lambda(x) = \sum \lambda_i x^{i-1} \left(\rho(x) = \sum \rho_i x^{i-1}\right) \qquad (7)$$

where $\lambda_i(\rho_i)$ is the fraction of edges incident to variable (check) nodes of degree i [37]. A parity-check matrix of code length 10 bits is given by:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix},$$

and Fig. 8 illustrates the corresponding Tanner graph representation [45, 46]. The parity-check matrix H is used to decode the received code sequence.

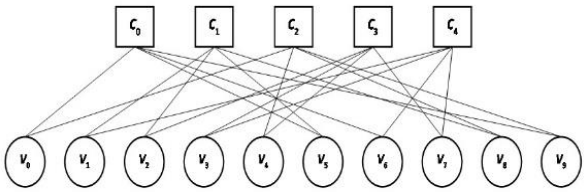

Fig. 8. Tanner graph representation of parity check matrix H

The codebook C is the set of length N words x, which satisfy:

$$\boldsymbol{x}.H^t = 0. \quad (8)$$

LDPC codes are popularly categorized into regular and irregular codes, random and pseudo-random codes, and structured and unstructured codes. Regular LDPC codes are commonly used, but carefully constructed irregular codes can also exhibit efficient error-correcting performance. LDPC block codes are similar to other block codes but are distinguished by the extended code length that introduces execution complexity and latency [34, 47].

Efficient LDPC decoding algorithms are designed to meet the cost, time, power and bandwidth requirements of intended applications. The construction of efficient LDPC codes involves addressing issues such as complexity, memory and latency. Achieving capacity-approaching and energy-efficient LDPC codes require not only the design of efficient decoders but also robust puncturing algorithms that allow for easy adjustment of block length and code rate [42, 48]. There are various techniques proposed in the literature for LDPC code construction and algorithmic optimization of decoding to address issues such as complexity, memory and latency. Efficient decoder construction aims to achieve good error correction performance and low error floor performance, which remains a challenging and active area of research for digital communication applications.

Efficiently constructed LDPC codes must not only achieve good error correction performance but also combat the degradation of error-floor performance in the high signal-to-noise ratio (SNR) region. This is typically achieved by utilizing a more dominant error-prone structure (EPS) than the low-weight codewords in the error floor region [43]. Having a general framework to determine the minimum distance and the minimum error-prone structures is crucial in designing an efficient LDPC code scheme. Techniques such as code matrix permutation, matrix space restriction, and sub-matrix row-column scheduling are commonly investigated in the literature to address the issue of decoder latency. These techniques aim to optimize the decoding process and reduce latency in LDPC code decoding, which is an important consideration in practical communication systems. It is however challenging to obtain accurate channel state information. The complex mathematical representation of the design models also lacks precise accuracy, leading to performance degradation.

It has been shown in [49] that a linear block code C of length N, with a cycle-free Tanner graph, does not support good codes. Therefore, a quasi-cyclic (QC) structure is generally imposed on the parity-check matrices of LDPC codes for efficient hardware implementation, as reported [34, 42, 44]. This implementation allows for easy adjustment of the block length, providing advantages such as a fast-decoding convergence rate and improvement on the error-floor problem. QC-LDPC codes must satisfy the condition that no two rows or columns of the parity-check matrices have more than one position where they both have non-zero components.

Costantini *et al.* [50] presented constructions of non-binary LDPC codes based on ultra-sparse matrices. According to the authors, non-binary LDPC codes over non-binary Galois fields (GFs) outperform binary LDPC codes by at least 0.3 dB. However, this performance gain comes at the cost of increased decoding complexity.

In [49], two possible approaches for decoding LDPC convolutional codes, namely block-wise decoding and windowed decoding, were discussed. The block-wise decoding technique starts the decoding process only when the entire codeword has been received, while the windowed decoder operates on a window of size W that slides along the Tanner graph sequentially. A summary of LDPC codes and their applications can be found in Table III.

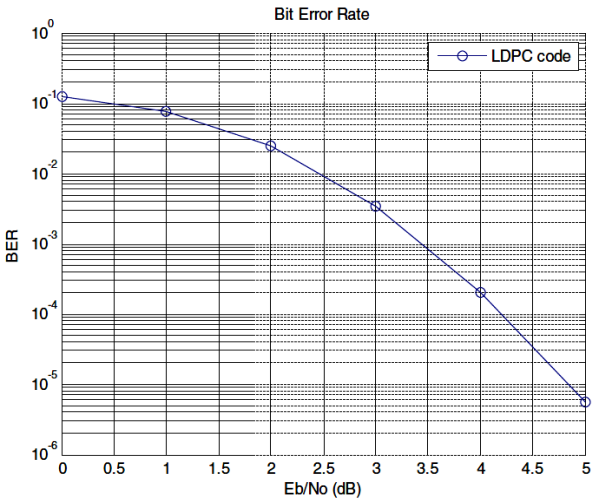

Fig. 9. Performance of LDPC in BER.

Fig. 9 depicts the simulation results of the BER performance of LDPC codes simulation in [8]. The simulation conditions are same as in Fig. 7; a code rate R of ½, over an additive white Gaussian noise (AWGN)

channel and binary phase shift keying (BPSK) modulation scheme. The use of LDPC codes and 3-dimensional turbo codes is integrated with receive diversity methods to serve as the error correction strategy over AWGN channels, employing a BPSK modulation scheme. A comparison of the BER performance of turbo codes and LDPC codes, as shown in Figs. 7–9, [8, 51, 52] indicates that LDPC codes offer better BER performance.

TABLE III: A SUMMARY OF LDPC CODES

Reference	Proposed LDPC	Methods	Results	Problem
Kim *et al.* [36]	Efficiently-encodable rate-compatible (E2RC) LDPC codes	Strategies for increasing the maximum puncturing rate.	Good performance at a moderately high puncturing rate.	Higher puncturing rates are difficult to achieve limiting performance.
Sariduman *et al.* [43]	Integer programming IP-based search technique for error-prone structures of LDPC codes	An iterative integer programming algorithm was proposed to enumerate all EPS parity-check matrices to find the important parameters of an LDPC code.	With the knowledge of the dominant EPS, the error-floor performance of LDPC codes can be estimated.	Proposed integer programming is not very efficient in finding minimum distance and minimum SS size for large block lengths above 1000.
Chaibi *et al.* [53]	Parallel genetic algorithm for LDPC codes	Parallel genetic algorithms (PGAD) for decoding low-density parity using multi-criteria optimization method.	Offers good performance when solving a complex optimization problem. Large gains over the sum-product decoder.	Increased decoding complexity.
Roth *et al.* and Zhu *et al.* [42][44]	Quasi-LPDC decoder	Constructed QC-LDPC codes from isomorphism theory.	Good performance under iterative decoding.	Algebraic and usually results in codes with regular degree distributions

III. DEEP LEARNING CHANNEL CODING

In [54], it is reported that early research in neural network applications in digital communication focused more on learning decoders rather than encoders, as the latter is more challenging in the specific problems they must solve. Decoder training focuses on error propagation, complexity matching, robustness of the channel model, feedback mechanisms and coordination of the learning rate. Whereas encoder training frequently concentrates on non-linearity, quantization, latency and channel variability. However, today ML algorithms are being introduced to achieve optimal end-to-end performance in communication systems. Nevertheless, it remains uncertain to what extent ML can replace or complement the domain expertise developed over the last century in communication [55].

Recently, there have been increasing attention on the application of ML in channel encoding schemes, particularly involving deep neural network autoencoder architecture, which can improve or provide solutions that are not achievable with conventional coding techniques [56, 57]. Deep learning (DL) has also been introduced in digital communication applications to address channel estimation problems [58, 59]. DL autoencoders have been applied in real-world communication such as natural language processing (text compression), image compression and transmission (JPEG AI), speech and audio compression (Google's WaveNet), satellite communication and image reconstruction (satellite and remote sensing) and recommendation systems (Netflix recommendation system [60, 61]. Autoencoders, being a type of deep learning scheme, do not heavily rely on heuristics of channel estimation, which allows them to adapt to communicate over any channel, even for which no information-theoretically optimal schemes are known. Autoencoders have been shown to understand the dynamics of end-to-end performance of the entire communication system building blocks, as reported in [62]. It has been established in [63] that DL-based communication systems, trained to optimize end-to-end performance, act as universal function approximators with superior algorithmic learning ability, even under complex channel conditions. Apart from the type of autoencoder used in Fig. 10, autoencoders have been used in a communication system, and different configurations and alternative autoencoders that have been investigated to enhance modulation, data compression, and error correction [64]. Some prominent modifications and alternative autoencoder components used are variational autoencoder, sparse autoencoder and denoising autoencoder [65, 66]. However, Fig. 10 illustrates an end-to-end autoencoder communication system [54] .

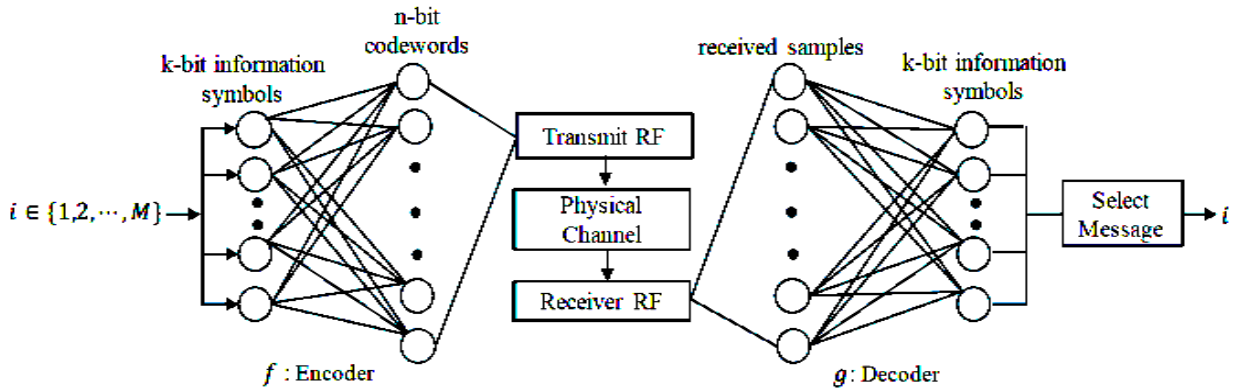

Fig. 10. Representation of a channel autoencoder.

The current research trend in autoencoder schemes involves training both the encoder and decoder with noisy feedback [15]. The autoencoder is trained to produce an efficient signal vector at the encoder, which can be efficiently decoded to retrieve the original message signal error-free, despite the presence of regularizers. Gaussian noise and dropout are commonly used as regularizers within the autoencoder model. Proper hyper-parameter selection is crucial for setting up an efficient DL autoencoder model, and an efficient training regime is equally important.

In [15], the authors proposed a turbo-encoded autoencoder, parameterized as convolutional neural networks with interleavers, inspired by turbo codes. To address the issue of locally optimal solutions in their encoder/decoder, they proposed training strategies involving alternate training of the encoder and decoder. The authors in [67] also support this concept of systematic training for their TurboNet decoder, which uses the max-log-MAP decoding algorithm. In [54], a two-step suboptimal training policy for the autoencoder with the Adam optimizer has been proposed.

The significance of a proper training regime cannot be overstated, as it has been reported that a perfectly trained autoencoder can adapt to any environment and obtain optimal codes for any block length. Authors generally recommend a large batch size for separate systematic encoder and decoder training, as it allows for easy convergence and optimal performance, as shown in experiments conducted in [15, 68].

Numerous studies, such as [15, 54, 62, 63, 69, 70, 71], have demonstrated the superior performance of DL communication applications over traditional codes and channel estimation under non-linear system models. The neural network architecture used in the deep learning autoencoder in this study are convolutional neural networks (CNNs) and recurrent neural networks (RNNs). RNNs are used for sequential data analysis and CNNs are widely used for image recognition [72]. The review of performances is summarized in Table VI, while Tables IV and V provide a selection list of commonly used hyper-parameters in machine learning applications. It has been observed that the sigmoid activation function, Adam optimizer and binary cross-entropy loss functions are commonly used in these applications.

TABLE IV: LOSS FUNCTIONS

Reference	Name	$l(\mathbf{u},\mathbf{v})$
Shinde and Shah [60]	MSE	$\|\mathrm{u}-\mathrm{v}\|_2^2$
Douillard *et al.*, Sattiraju *et al.*, Nachmani *et al.*, Soltani *et al.* [14, 31, 59, 62]	Binary cross-entropy	$-\sum_j u_j \log(v_j)$

TABLE V: ACTIVATION FUNCTIONS

Reference	Name	$[\boldsymbol{\sigma}(\mathbf{u})]_i$	Range
Sattiraju *et al.* [31]	linear	u_i	$(-\infty,\infty)$
Nachmani *et al.* [52]	ReLU	$\max(0,u_i)$	$[\mathbf{0},\infty)$
Nachmani *et al.* and Liu *et al.* [52, 60]	tanh	$\tanh(u_i)$	$(-1,1)$
Sattiraju *et al.*, Nachmani *et al.* Nachmani *et al.* [31, 52, 59]	sigmoid	$\frac{\mathbf{1}}{\mathbf{1}+e^{-u_i}}$	$(\mathbf{0},\mathbf{1})$
Liu *et al.* [60]	softmax	$\frac{e^{u_i}}{\sum_j e^{u_j}}$	(0,1)

IV. LESSONS LEARNT AND FUTURE RESEARCH DIRECTION

In this survey, we learnt that early research in neural network applications in digital communication focused on learning decoders rather than encoders, which was found to be more challenging. Also, ML algorithms, such as DL autoencoders, have been introduced to achieve optimal end-to-end performance in communication systems, surpassing traditional coding techniques. Autoencoders improve performance robustly by adjusting to channel circumstances and variations, adapting to difficult settings to improve error correction. It also allows more effective use of resources by lowering overhead and consequently, optimizing the overall communication system.

ML applications produce effective representation for robust signal recovery by learning from real-world changing circumstances in non-linear system models [73].

Furthermore, proper hyper-parameter selection and training regimes are crucial for the performance of DL autoencoders, with large batch sizes and systematic encoder/decoder training strategies being recommended for optimal convergence and performance.

For future research directions, the in-depth exploration of DL autoencoders and their applications in digital

communication is still needed, with a focus on improving the efficiency and effectiveness of encoder training. Latency issues must also be looked at, especially in some time-critical applications. Also, the investigation of novel regularization techniques and hyper-parameter selection methods that enhance the performance of autoencoder-based communication systems are still needed from the research community. Furthermore, research that explores various autoencoder architectures beyond CNNs and interleavers, to identify other potential approaches for optimizing end-to-end performance is also necessary. There is also a need to carry out comparative studies of different DL communication applications and traditional coding techniques under various system models and channel conditions, to better understand the advantages and limitations of each approach. Lastly, investigations of the potential of transfer learning and adaptation techniques to improve their performance in different communication scenarios and environments are still needed (

TABLE VI: PERFORMANCE SUMMARY OF DEEP LEARNING APPLICATIONS IN CHANNEL CODING AND ESTIMATION

Ref.	Proposed Technique	Approach	Results
Jiang *et al.* [15]	Turbo autoencoder	introducing Turbo Autoencoder using the neural structure and training algorithms.	Outperform traditional codes in the low to middle SNR range. Performance is only worse than LDPC and polar code.
He *et al.* [67]	TurboNet: Deep neural network turbo code decoder	for turbo decoding that integrates DL into the traditional max-log-maximum	Superiority in error-correction, ability, signal-to-noise ratio generalization.
Devamane and Itagi [70]	Recurrent neural network (RNN) turbo decoder	Turbo decoders are constructed by two means; neural Turbo decoder and deep learning Turbo decoder. The performances are examined	It is concluded that both neural and DL turbo decoders perform better as compared to conventional Viterbi decoders.
Balevi and Andrews [54]	Autoencoder ECC	Using a concatenation of a turbo code and an autoencoder. Utilizing turbo codes as an implicit regularization. Suboptimum training methods adopted.	The proposed coding technique outperforms conventional turbo codes for one-bit quantization.
Soltani *et al.* [62]	Autoencoder-based optical wireless communications systems (OWS)	Proposed autoencoder based OWC.	Performance compared with the state-of-the-art model-based OWC systems in terms of the block error rate (BLER) metric. Results indicate learning-based OWC system outperforms the model-based one.
Mei *et al.* [71]	Machine learning-based channel estimation in OFDM	Simulated the performance of machine learning-based channel estimation under quasi-stationary channel conditions.	Simulation results exhibit the effectiveness and convenience of machine learning-based channel estimation under complex channel models.
Nachmani *et al.* [68]	Novel deep learning to improve belief propagation algorithm	Used “soft” Tanner instead of the standard Tanner graph. Properly weighting messages, such that the effect of small cycles in the Tanner graph was partially compensated.	Improved BER performance.
Sattiraju *et al.* [62]	RNN turbo encoder/decoder	Used the LTE variant of the turbo encoder to encode the data.	Their testing accuracy produces 67% for turbo encoding and 100% for turbo decoding.

IV. CONCLUSION

We conducted a comparative analysis of near-capacity turbo codes and LDPC codes, highlighting their potential and challenges. The traditional approach of designing communication algorithms based on complex mathematical models and heuristics was found to be sub-optimal. They are characterized by a lack of accurate model estimation. We also discussed the emergence of ML applications in communication systems as a potential solution to the limitations of conventional coding schemes.

Our survey revealed that while LDPC codes may offer better performance than turbo codes, ML-based approaches have shown competitive performance and adaptability to channel conditions. The superiority of ML-based end-to-end channel coding and estimation in non-linear system models has been demonstrated in several reviewed papers. We found that systematic training schedules are generally favored in ML applications for communication to prevent overfitting and local optima issues. It has been established that a perfectly trained autoencoder can dynamically adapt to channel interference to find optimal error correction codes, although achieving perfect training remains a challenge.

However, it should be noted that the theoretical understanding of ML deep neural networks is still unsatisfactory, and there may be a need to open the “black boxes” of ML to fully comprehend and harness their full potential. We believe that ML applications will eventually replace communication algorithms with learned weights derived from training deep neural network architectures, eliminating the need for specialized expert channel coding schemes that create unnecessary bottlenecks.

CONFLICT OF INTEREST

. The authors declare no conflict of interest

AUTHOR CONTRIBUTIONS

Kayode Olaniyi conceptualized and carried out the research. Reolyn Heymann supervised the research. Theo Swart helped with the data analysis and the result discussion section. All authors approved the final version.

REFERENCES

[1] S. Shao *et al.*, "Survey of Turbo, LDPC, and Polar Decoder ASIC Implementations," *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, pp. 2309–2333, 2019, doi: 10.1109/COMST.2019.2893851.

[2] G. Battail, "A conceptual framework for understanding turbo codes," *IEEE J. Sel. Areas Commun.*, vol. 16, no. 2, pp. 245–254, 1998, doi: 10.1109/49.661112.

[3] C. Berrou, R. Pyndiah, P. Adde, C. Douillard, and R. Le Bidan, "An overview of turbo codes and their applications," *Wirel. Technol. 2005, Conf. Proc. - 8th Eur. Conf. Wirel. Technol.*, vol. 2005, pp. 1–10, 2005, doi: 10.1109/ecwt.2005.1617639.

[4] K. Gracie and M.-H. Hamon, "Turbo and turbo-like codes: Principles and applications in telecommunications," *Proc. IEEE*, vol. 95, no. 6, pp. 1228–1254, 2007.

[5] J. D. Kene and K. D. Kulat, "Soft Output Decoding Algorithm for Turbo Codes Implementation in Mobile Wi-Max Environment," *Procedia Technol.*, vol. 6, pp. 666–673, 2012, doi: 10.1016/j.protcy.2012.10.080.

[6] K. Ramasamy, B. Balakrishnan, and M. U. Siddiqi, "A new class of asymmetric turbo code for 3G systems," *AEU - Int. J. Electron. Commun.*, vol. 60, no. 6, pp. 447–458, 2006, doi: 10.1016/j.aeue.2005.09.007.

[7] B. Vucetic, Y. Li, L. C. Perez, and F. Jiang, "Recent advances in turbo code design and theory," *Proc. IEEE*, vol. 95, no. 6, pp. 1323–1344, 2007, doi: 10.1109/JPROC.2007.897975.

[8] M. Mohammed, A. Abdessadek, and E. H. Ali, "A comparative simulation study on the performance of LDPC codes and 3Dimensional turbo codes," *Lect. Notes Networks Syst.*, vol. 25, pp. 21–35, 2018, doi: 10.1007/978-3-319-69137-4_3.

[9] Y. Yang and Y. Li, "Research and Implementation of Turbo Coding Technology in High-Speed Underwater Acoustic OFDM Communication," *J. Robot.*, vol. 2022, 2022.

[10] M. Arif, N. M. Sheikh, and A. U. H. Sheikh, "Design of two step deterministic interleaver for turbo codes," *Comput. Electr. Eng.*, vol. 34, no. 5, pp. 368–377, 2008, doi: 10.1016/j.compeleceng.2007.10.009.

[11] T. Matsumine and H. Ochiai, "Capacity-Approaching Non-Binary Turbo Codes: A Hybrid Design Based on EXIT Charts and Union Bounds," *IEEE Access*, vol. 6, pp. 70952–70963, 2018, doi: 10.1109/ACCESS.2018.2881243.

[12] T. Tonnellier, C. Leroux, B. Le Gal, B. Gadat, C. Jego, and N. Van Wambeke, "Lowering the Error Floor of Turbo Codes with CRC Verification," *IEEE Wirel. Commun. Lett.*, vol. 5, no. 4, pp. 404–407, 2016, doi: 10.1109/LWC.2016.2571283.

[13] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near SHANNON limit error-correcting coding and encoding: Turbo-codes (1)," *IEEE Int. Conf. Commun.*, no. 1, pp. 1064–1070, 1993, doi: 10.1109/icc.1993.397441.

[14] C. Douillard and M. Jezequel, "Turbo Codes: From First Principles to Recent Standards," *Channel Coding Theory, Algorithms, Appl. Acad. Press Libr. Mob. Wirel. Commun.*, pp. 1–52, 2014, doi: 10.1016/B978-0-12-396499-1.00001-7.

[15] Y. Jiang, S. Kannan, H. Kim, S. Oh, H. Asnani, and P. Viswanath, "Turbo Autoencoder: Deep learning based channel codes for point-to-point communication channels," *Adv. Neural Inf. Process. Syst.*, vol. 32, no. NeurIPS, pp. 1–11, 2019.

[16] A. S. Babu and M. A. Ambroze, "From Convolutional Codes to Turbo Codes," *Adv. Commun. Electron. Networks, Robot. Secur. Vol. 13*, vol. 13, p. 105, 2016.

[17] H. Jin and R. J. McEliece, "Coding theorems for turbo code ensembles," *IEEE Trans. Inf. Theory*, vol. 48, no. 6, pp. 1451–1461, 2002, doi: 10.1109/TIT.2002.1003833.

[18] R. Garzón Bohórquez, C. A. Nour, and C. Douillard, "On the equivalence of interleavers for turbo codes," *IEEE Wirel. Commun. Lett.*, vol. 4, no. 1, pp. 58–61, 2015, doi: 10.1109/LWC.2014.2367517.

[19] G. Matz and F. Hlawatsch, "Fundamentals of time-varying communication channels," in *Wireless communications over rapidly time-varying channels*, Elsevier, 2011, pp. 1–63.

[20] G. R. Rao and G. S. Rao, "Performance analysis of 64QAM Turbo coded OFDM for 4G applications," *Procedia Comput. Sci.*, vol. 143, pp. 907–913, 2018, doi: 10.1016/j.procs.2018.10.363.

[21] S. V Zaitsev, V. V Kazymyr, V. M. Vasilenko, and A. V Yarilovets, "Adaptive selection of parameters of s-random interleaver in wireless data transmission systems with turbo coding," *Radioelectron. Commun. Syst.*, vol. 61, pp. 13–21, 2018.

[22] R. Garzón Bohórquez, C. A. Nour, and C. Douillard, "Protograph-Based Interleavers for Punctured Turbo Codes," *IEEE Trans. Commun.*, vol. 66, no. 5, pp. 1833–1844, 2018, doi: 10.1109/TCOMM.2017.2783971.

[23] O. Işcan and W. Xu, "Window-Interleaved Turbo Codes," *IEEE Commun. Lett.*, vol. 22, no. 4, pp. 676–679, 2018, doi: 10.1109/LCOMM.2018.2794359.

[24] K. Bhowmik, "A Review of Turbo Decoder for Wireless Communication System through VLSI Design," vol. 7, no. 05, pp. 451–455, 2018.

[25] A. H. Mugaibel and M. A. Kousa, "Turbo Codes: Promises and Challenges," *King Fahd Univ. Pet. Miner. PO Box*, vol. 1721, 1999.

[26] P. Pfeifer and H. T. Vierhaus, "Reconfiguration Aspects of PENCA - An Area-efficient Reconfigurable Encoder Architecture with Built-in Security Features for Flexible Error Detection and Correction in Robust Dependable Communication Systems," *IFAC-PapersOnLine*, vol. 49, no. 25, pp. 390–395, 2016, doi: 10.1016/j.ifacol.2016.12.076.

[27] S. D. B and I. R.L., "A Review on Punctured Analysis of Turbo codes," *Ijireeice*, vol. 3, no. 12, pp. 140–142, 2015, doi: 10.17148/ijireeice.2015.31229.

[28] A. Mirza and S. A. Sheikh, "Performance Comparison of Turbo Decoding Algorithms," *Can. J. Signal Process.*, vol. 1, no. 2, July, 2010.

[29] C. Studer, C. Benkeser, S. Belfanti, and Q. Huang, "Design and implementation of a parallel turbo-decoder ASIC for 3GPP-LTE," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 8–17, 2010.

[30] Y. He, J. Zhang, S. Member, S. Jin, and S. Member, "Model-Driven DNN Decoder for Turbo Codes : Design , Simulation and Experimental Results," vol. 6778, no. c, pp. 1–16, 2020, doi: 10.1109/TCOMM.2020.3010964.

[31] R. Sattiraju, A. Weinand, and H. D. Schotten, "Performance Analysis of Deep Learning based on Recurrent Neural Networks for Channel Coding," *2018 IEEE Int. Conf. Adv. Networks Telecommun. Syst.*, pp. 1–6, 2018, doi: 10.1109/ANTS.2018.8710159.

[32] J. Sodha, "Turbo code frame synchronization," *Signal Processing*, vol. 82, no. 5, pp. 803–809, 2002, doi: 10.1016/S0165-1684(02)00159-7.

[33] C. Tanriover, B. Honary, J. Xu, and S. Lin, "Improving turbo code error performance by multifold coding," *IEEE Commun. Lett.*, vol. 6, no. 5, pp. 193–195, 2002, doi: 10.1109/4234.1001661.

[34] C. Panem, V. R. Gad, and R. S. Gad, "Sensor's data transmission with BPSK using LDPC (Min-Sum) error corrections over MIMO channel: Analysis over RMSE and BER," *Mater. Today Proc.*, vol. 27, no. xxxx, pp. 571–575, 2020, doi: 10.1016/j.matpr.2019.12.039.

[35] D. J. C. MacKay and R. M. Neal, "Near Shannon limit performance of low density parity check codes," *Electron. Lett.*, vol. 33, no. 6, pp. 457–458, 1997.

[36] J. Kim, W. Hur, A. Ramamoorthy, and S. W. McLaughlin, "Design of rate-compatible irregular LDPC codes for incremental redundancy hybrid ARQ systems," *IEEE Int. Symp. Inf. Theory - Proc.*, pp. 1139–1143, 2006, doi: 10.1109/ISIT.2006.261962.

[37] W. Ullah and A. Yahya, "Comprehensive Algorithmic Review and Analysis of LDPC Codes," *TELKOMNIKA Indones. J. Electr. Eng.*, vol. 16, no. 1, p. 111, 2015, doi: 10.11591/tijee.v16i1.1595.

[38] I. Develi and Y. Kabalci, "A comparative simulation study on the performance of LDPC coded communication systems over Weibull fading channels," *J. Appl. Res. Technol.*, vol. 14, no. 2, pp. 101–107, 2016.

[39] H. Saeedi and A. H. Banihashemi, "Design of irregular LDPC codes for BIAWGN channels with SNR mismatch," *IEEE Trans. Commun.*, vol. 57, no. 1, pp. 6–11, 2009.

[40] N. Andreadou, C. Assimakopoulos, and F.-N. Pavlidou, "Performance evaluation of LDPC codes on PLC channel compared to other coding schemes," in *2007 IEEE Int. Symp. on Power Line Commun. and Its Appl.*, 2007, pp. 296–301.

[41] H. Wei and A. H. Banihashemi, "ADMM check node penalized decoders for LDPC codes," *IEEE Trans. Commun.*, vol. 69, no. 6, pp. 3528–3540, 2021.

[42] C. Roth, P. Meinerzhagen, C. Studer, and A. Burg, "A 15.8 pJ/bit/iter Quasi-Cyclic LDPC Decoder for IEEE 802.11n in 90 nm CMOS," *IEEE Asian Solid-State Circuits Conf.*, pp. 8–11, 2010.

[43] A. Sariduman, A. E. Pusane, and Z. C. Taşkin, "An integer programming-based search technique for error-prone structures of LDPC codes," *AEU - Int. J. Electron. Commun.*, vol. 68, no. 11, pp.

1097–1105, 2014, doi: 10.1016/j.aeue.2014.05.012.
[44] H. Zhu, B. Zhang, M. Xu, H. Li, and H. Xu, "Array based quasi-cyclic LDPC codes and their tight lower bounds on the lifting degree," *Phys. Commun.*, vol. 36, p. 100765, 2019, doi: 10.1016/j.phycom.2019.100765.
[45] V. A. Chandrasetty and S. M. Aziz, "Overview of LDPC codes," *Resour. Effic. LDPC Decod.*, pp. 5–10, 2018, doi: 10.1016/b978-0-12-811255-7.00002-2.
[46] W. E. Ryan, "An introduction to LDPC codes," *CRC Handb. Coding Signal Process. Rec. Syst.*, vol. 5, no. 2, pp. 1–23, 2004.
[47] L. Mostari and A. Taleb-Ahmed, "High performance short-block binary regular LDPC codes," *Alexandria Eng. J.*, vol. 57, no. 4, pp. 2633–2639, 2018, doi: 10.1016/j.aej.2017.09.016.
[48] J. Kim, A. Ramamoorthy, and S. W. McLaughlin, "Design of efficiently-encodable rate-compatible irregular LDPC codes," *IEEE Int. Conf. Commun.*, vol. 3, no. c, pp. 1131–1136, 2006, doi: 10.1109/ICC.2006.254899.
[49] H. Ben Thameur, B. Le Gal, N. Khouja, F. Tlili, and C. Jego, "A survey on decoding schedules of LDPC convolutional codes and associated hardware architectures," *Proc. - IEEE Symp. Comput. Commun.*, pp. 898–905, 2017, doi: 10.1109/ISCC.2017.8024640.
[50] L. Costantini, B. Matuz, G. Liva, E. Paolini, and M. Chiani, "On the performance of moderate-length non-binary LDPC codes for space communications," *ASMS/SPSC 2010 2010 5th Adv. Satell. Multimed. Syst. Conf. 11th Signal Process. Sp. Commun. Work.*, pp. 122–126, 2010, doi: 10.1109/ASMS-SPSC.2010.5586886.
[51] Y. Liu, X. Liu, Z. Ding, Y. Hu, and L. Zhao, "A new LDPC decoding scheme based on BP and Gated Neural Network," in *2020 5th Int. Conf. on Info. Science, Comput. Tech. and Transportation (ISCTT)*, 2020, pp. 346–349.
[52] E. Nachmani, Y. Be'Ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *2016 54th Annual Allerton Conf. on Commun., Control, and Comput. (Allerton)*, 2017, pp. 341–346, doi: 10.1109/ALLERTON.2016.7852251.
[53] H. Chaibi, A. Chehri, R. Saadane, and A. Zimmerman, "Parallel genetic algorithm decoder scheme based on DP-LDPC codes for industrial IoT scenarios," *Procedia Comput. Sci.*, vol. 176, pp. 3496–3505, 2020, doi: 10.1016/j.procs.2020.09.047.
[54] E. Balevi and J. G. Andrews, "Autoencoder-Based Error Correction Coding for One-Bit Quantization," *IEEE Trans. Commun.*, vol. 68, no. 6, pp. 3440–3451, 2020, doi: 10.1109/TCOMM.2020.2977280.
[55] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," *2016 IEEE Int. Symp. Signal Process. Inf. Technol. ISSPIT 2016*, pp. 223–228, 2017, doi: 10.1109/ISSPIT.2016.7886039.
[56] T. O'Shea and J. Hoydis, "An Introduction to Deep Learning for the Physical Layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, 2017, doi: 10.1109/TCCN.2017.2758370.
[57] P. Li, Y. Pei, and J. Li, "A comprehensive survey on design and application of autoencoder in deep learning," *Appl. Soft Comput.*, p. 110176, 2023.
[58] A. M. Tonello, N. A. Letizia, D. Righini, and F. Marcuzzi, "Machine Learning Tips and Tricks for Power Line Communications," *IEEE Access*, vol. 7, pp. 82434–82452, 2019, doi: 10.1109/ACCESS.2019.2923321.
[59] E. Nachmani, E. Marciano, D. Burshtein, and Y. Be'ery, "RNN decoding of linear block codes," *arXiv Prepr. arXiv1702.07560*, 2017.
[60] S. Liu, T. Wang, and S. Wang, "Toward intelligent wireless communications: Deep learning - based physical layer technologies," *Digit. Commun. Networks*, vol. 7, no. 4, pp. 589–597, 2021, doi: 10.1016/j.dcan.2021.09.014.
[61] C. Shorten, T. M. Khoshgoftaar, and B. Furht, "Deep Learning applications for COVID-19," *J. big Data*, vol. 8, no. 1, pp. 1–54, 2021.
[62] M. Soltani, W. Fatnassi, A. Aboutaleb, Z. Rezki, A. Bhuyan, and P. Titus, "Autoencoder-Based Optical Wireless Communications Systems," *2018 IEEE Globecom Work. GC Wkshps 2018 - Proc.*, no. December, 2019, doi: 10.1109/GLOCOMW.2018.8644104.
[63] T. Wang, C. K. Wen, H. Wang, F. Gao, T. Jiang, and S. Jin, "Deep learning for wireless physical layer: Opportunities and challenges," *China Commun.*, vol. 14, no. 11, pp. 92–111, 2017, doi: 10.1109/CC.2017.8233654.
[64] S. Yu and J. C. Principe, "Understanding Autoencoders with Information Theoretic Concepts," no. October 2021, 2019, doi: 10.1016/j.neunet.2019.05.003.
[65] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv Prepr. arXiv1312.6114*, 2013.
[66] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proc. of the 25th int. conf. on Machine learning*, 2008, pp. 1096–1103.
[67] Y. He, J. Zhang, S. Jin, C. K. Wen, and G. Y. Li, "Model-Driven DNN Decoder for Turbo Codes: Design, Simulation, and Experimental Results," *IEEE Trans. Commun.*, vol. 68, no. 10, pp. 6127–6140, 2020, doi: 10.1109/TCOMM.2020.3010964.
[68] E. Nachmani, Y. Be'ery, and D. Burshtein, "Learning to decode linear codes using deep learning," in *2016 54th Annual Allerton Conf. on Commun., Control, and Comput. (Allerton)*, 2016, pp. 341–346.
[69] S. Krastanov and L. Jiang, "Deep Neural Network Probabilistic Decoder for Stabilizer Codes," *Sci. Rep.*, vol. 7, no. 1, pp. 1–7, 2017, doi: 10.1038/s41598-017-11266-1.
[70] S. B. Devamane and R. L. Itagi, "Recurrent neural network based turbo decoding algorithms for different code rates," *J. King Saud Univ. - Comput. Inf. Sci.*, no. xxxx, 2020, doi: 10.1016/j.jksuci.2020.03.012.
[71] K. Mei, J. Liu, X. Zhang, and J. Wei, "Machine Learning Based Channel Estimation: A Computational Approach for Universal Channel Conditions," 2019, [Online]. Available: http://arxiv.org/abs/1911.03886.
[72] L. Alzubaidi *et al.*, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *J. big Data*, vol. 8, pp. 1–74, 2021.
[73] D. Wang and M. Zhang, "Artificial intelligence in optical communications: from machine learning to deep learning," *Front. Commun. Networks*, vol. 2, p. 656786, 2021.