IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 26 April 2024, accepted 27 May 2024, date of publication 11 June 2024, date of current version 3 July 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3412192

TOPICAL REVIEW

A Review on Machine Learning for Channel Coding

HEIMRIH LIM MENG KEE[1], NORULHUSNA AHMAD[1], (Senior Member, IEEE), MOHD AZRI MOHD IZHAR[1], (Senior Member, IEEE), KHOIRUL ANWAR[2], (Senior Member, IEEE), AND SOON XIN NG[3], (Senior Member, IEEE)

[1]Ubiquitous Broadband Access Network Research Group, Faculty of Artificial Intelligence, Universiti Teknologi Malaysia, Kuala Lumpur 54100, Malaysia
[2]The University Center of Excellence for Advanced Intelligent Communications (AICOMS), Telkom University, Bandung 40257, Indonesia
[3]School of Electronics and Computer Science, University of Southampton, SO17 1BJ Southampton, U.K.

Corresponding author: Norulhusna Ahmad (norulhusna.kl@utm.my)

This work was supported in part by the Ministry of Higher Education through the Fundamental Research Grant Scheme under Grant FRGS/1/2021/ICT09/UTM/02/1; in part by the Huawei-Telkom University (TelU) Beyond 5.5G Laboratory, TelU, Bandung, Indonesia; and in part by the National Institute of Information and Communications Technology (NICT) through ICT Virtual Organization of ASEAN Institutes and NICT (ASEAN IVO) Project 2022 under Grant S.K130000.7656.4X796.

ABSTRACT The usage of artificial intelligence and machine learning in wireless communications is the stepping stone towards a technological breakthrough in the current limitations of wireless communication systems. The trend of future coding schemes towards 6G appears to be based on rateless schemes and machine learning. Channel coding is important when transmitting data or information reliably as it provides error-correcting purposes. However, there is still a demand for more research regarding machine learning for channel coding. There is also a lack of a specific term or classification for existing machine learning applications for channel coding. This paper explores and compiles current trending machine learning techniques for channel coding. We are also introducing and proposing a new type of machine learning classification for channel coding purposes, as well as surveying some of the papers that fall under the respective class. This paper also discusses current challenges and future machine learning trends for channel coding, which are expected to impact future wireless communications development, especially in channel coding advancements.

INDEX TERMS 6G, 5G advanced, wireless communications, artificial intelligence, channel coding, machine learning, deep learning, reinforcement learning, federated learning.

I. INTRODUCTION

The 3rd Generation Partnership Project (3GPP) announced the fifth generation-advanced (5G-A), which was started during Release 18, as the next step in developing 5G. This aims to enhance further 5G performance, flexible spectrum use, diverse 5G devices, evolved network topology and data-driven and AI-powered 5G. These main flagship or main goals for 5G-A are key to the evolution of 5G and will certainly make its path towards our goal of 6G in the future release [1]. A 3GPP timeline for 5G-A and 6G roadmap, as well as the overview of 5G-A, is shown in Fig. 1 based on [1] and [2].

Further enhanced 5G performance can be achieved by improving network energy efficiency, coverage, mobility support, multiple-input multiple-output (MIMO), multicast and broadcast services, and positioning. Studies are focused on reducing network energy consumption to achieve sustainability and combat climate change. Release 18 also aims to improve energy efficiency by studying techniques to reduce network energy consumption in targeted deployment scenarios. Another important consideration for further enhancement is coverage, which will be enhanced by studying techniques to extend uplink coverage. Another aim is to explore ways to improve power efficiency, reduce power reduction

The associate editor coordinating the review of this manuscript and approving it for publication was M. Shamim Kaiser.

and increase the power limit for user equipment. The improvement of conditional handover support, in which an instruction from the network for handover will be received by the user equipment, is another significant topic for Release-18 mobility support development. MIMO evolution will continue with 3GPP Release 18 by exploring potential CSI reporting enhancements and a larger number of orthogonal demodulation reference signal ports. The multicast and broadcast services will be extended to support user equipment in radio resource control inactive state and improve resource efficiency in radio access networks (RAN) sharing scenarios. Finally, positioning solutions will be investigated to improve accuracy, integrity, and power efficiency for RedCap devices.

The key enabling features of flexible spectrum use in the 3GPP Release 18 includes plans to support 5G deployments in spectrum allocations less than 5 MHz, improve dynamic spectrum sharing performance by increasing physical downlink control channel capacity, allow for efficient use of fragmented spectrum blocks, study the feasibility of co-existing downlink and uplink within a time division duplex band, and handle cross-link interference better for dynamic time division duplex in commercial deployments.

In the context of enhanced Mobile BroadBand (eMBB), ultra-reliable low latency communications (URLLC), and massive Machine Type Communications (mMTC) usage scenarios, 5G must support a variety of device kinds. To better support smartphones as well as other 5G devices, such as extended reality (XR) and cloud gaming devices, low-complexity user equipment, vehicular devices, and unmanned aerial aircraft (UAV), 3GPP Release 18 will continue to research and provide specialised functionality. The goal is to improve and expand 5G connectivity for a range of gadgets and uses. This is for user equipment devices, multi-SIM devices, XR and cloud gaming devices, RedCap devices, and devices for low-power transmission, thus enhancing in-device coexistence (IDC). Additionally, Release 18 will improve sidelink connectivity for devices mounted on automobiles and aerial aircraft. Increased data rate, decreased latency, less complexity, decreased power consumption, and support for coexistence with long-term evolution (LTE) and New Radio (NR) devices are among Release 18's primary goals. Additionally, Release 18 will have 5G NR compatibility for UAVs.

The aim of achieving an evolved network topology is to increase the split next-generation RAN (NG-RAN) architecture's resilience; 3GPP Release 18 makes use of a variety of nodes, including integrated access and backhaul nodes, radio frequency repeaters, relays, and interaction with non-terrestrial networks. The splitting of a 5G Node B into two pieces is supported by the next-generation RAN, also known as the RAN for 5G. The study will emphasise strengthening the control plane's resilience of the centralised unit and expanding the functionality of network-controlled repeaters and integrated access and backhaul nodes.

To better manage the increasingly complex 5G networks, 3GPP Release 18 will use artificial intelligence (AI) and machine learning (ML) technology. It will improve data-collecting capabilities and investigate the use of AI to air interface operations. In research on AI-enabled RAN, Release 17 identified methods for network energy conservation, load balancing, and mobility optimization. Release 18 will improve the NR QoE management framework for future 5G services and handle SON/MDT data collecting. To enhance performance and lower complexity/overhead, it will also research AI/ML for the NR air interface and concentrate on specific use cases like CSI feedback, beam management, and location.

We can see that 5G Advanced will use AI and ML as it is one of the main flagships in Release 18. ML is an advancement of AI, also known as a subset of AI, which is eventually bound to happen. It was suggested as early as 1950 by Alan Turing, and it is known as the Turing test. He suggested that if a machine is capable of proving to humans that the machine is also human, the machine is considered "intelligent" [3]. In 1959, Arthur Samuel wrote a program that can play checkers on its own and improve itself [4]. In 1967, the development of the nearest neighbour algorithm opened up a new breakthrough for computers to achieve basic pattern recognition to solve problems such as the travelling salesman problem [5]. In 1979, the backbone of all modern Deep Learning (DL), which is the Artificial Neural Network (ANN), was introduced by Kunihiko Fukushima for his works on neocognitron [6]. An early version of Recurrent Neural Network(RNN), which is the Hopfield network, was popularized in 1982 [7]. In the 1990s, there was a lot of surges in ML development, such as the TD-Gammon based on ANN [8], which beats skilled players and also the famous Deep Blue project by IBM [9], which defeated Garry Kasparov, world champion of chess at the time. Other than that, there was also the development of Support Vector Machines(SVM) [10] and LSTM Recurrent Neural Networks [11], which are among the current techniques in ML these days. At the end of this decade, the famous dataset, MNIST (Modified National Institute of Standards and Technology) [12], which is a widely recognised benchmark of dataset until this day, was released.

The 2000s is also important as the term Deep learning was introduced by Hinton et al. [13]. This led to further advancement of the recognition of text and images with DL algorithms. Another famous database known as ImageNet was also developed by Deng et al. [14]. This database aims to help in image recognition development. Further development of the ImageNet introduces ImageNet Classification [15], which is a model that largely improves the performance of ML and image recognition. In 2016, Google created an AlphaGo program which is the first computer program to beat a professional human player using ML techniques [16].

AI has grown immensely over the years and impacted many fields, providing a paradigm shift to existing technologies. In the field of wireless communications, AI has been implemented in various physical layer and network layer

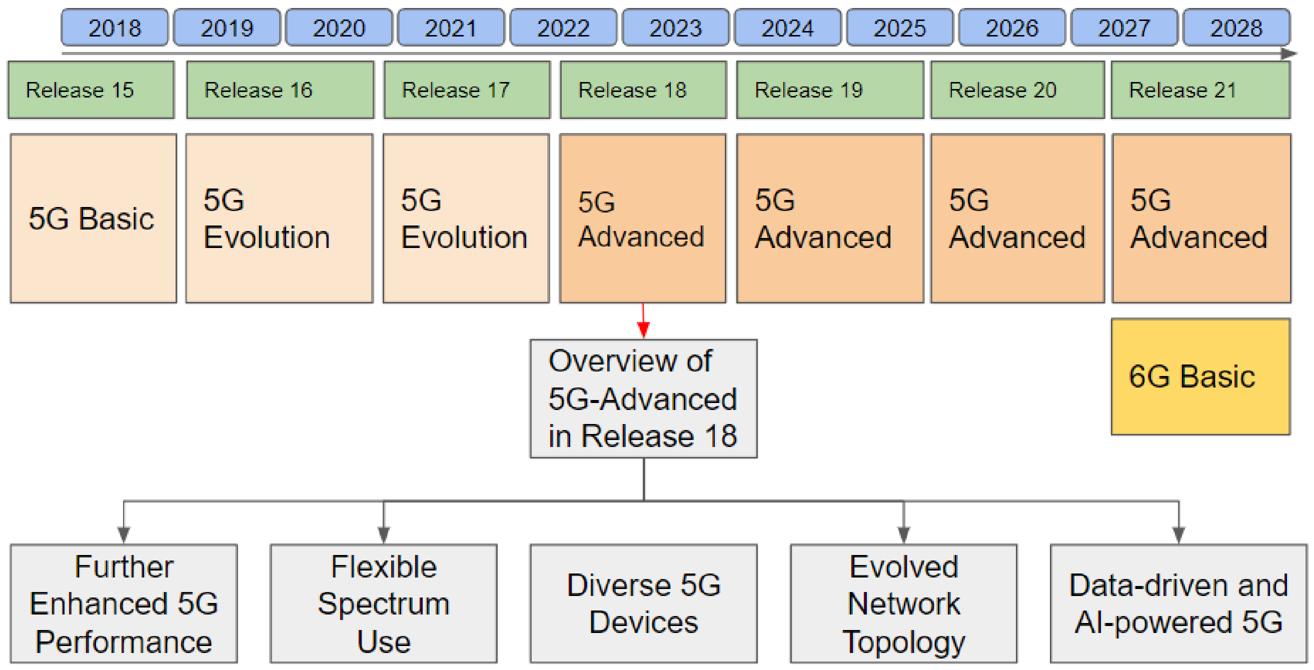

FIGURE 1. **A timeline of 3GPP for 5G to 6G (indicative).**

applications. Specifically, ML algorithms have been used for different layers of wireless networks, such as traffic clustering and resource allocation strategies [17].

Network management benefited a lot from ML research. Resource management problems are often improved using reinforcement learning, such as in [18] and [19]. Other than that, network security and quality of experience optimization also benefitted from ML research in wireless communications [20], [21]. An autoencoder for intruder detection was also found to be able to perform better using ML algorithm [22]. Reinforcement learning for routing protocol design in underwater sensor networks was studied in [23] to achieve an energy-efficient scheme. Numerous studies have been conducted on AI for the wireless network layer, including those on traffic flow prediction [24], and traffic congestion control [25] utilising autoencoders and K-means clustering, respectively.

In the physical layer, deep learning for channel estimation and symbol detection has been conducted in [26]. Authors in [27] also show that using deep learning for symbol detection can achieve good performance with low complexity in the design. Other deep learning application also includes channel estimation accuracy in [28] and the development of autoencoders in [29], which are able to reconstruct the signals from an impaired channel.

Current 5G standards in the case of eMBB employed Low-Density Parity-Check codes as standard. High throughput is required by 5G deployment scenarios, particularly the eMBB case. To manage this, the encoding and decoding of 5G channel codes specifically for data has to be created. 5G low-density parity check (LDPC) codes use a quasi-cyclic (QC) LDPC coding structure [30]. However, there is a potential to improve the channel coding aspect of wireless communication using ML. This remaining problem in wireless communication needs to be addressed as it will improve the bit error rate (BER) for any transmitted signal through an efficient channel coding scheme.

One of the challenges of 5G includes achieving a short packet with high-reliability and low-latency service [31]. 6G wireless communication networks will need significant paradigm changes to address these shortcomings of 5G [32]. All spectra, including the sub-6 GHz, mmWave, THz, and optical frequency bands, will be thoroughly investigated in order to provide a better data rate [33]. AI technologies will effectively be coupled with 6G wireless communication networks to provide comprehensive applications for enhanced network automation and management [34]. The development of 6G wireless communication networks will be significantly aided by Internet of Things (IoT), energy-efficient wireless network control, and federated learning systems.

Table 1 is a compilation of existing research and surveys on AI for wireless communication systems. Articles in [35], [36], [37], [38], and [39] reviewed the techniques of AI in wireless communication without focusing on the implementation of the channel coding compared to the articles in [40], [41], [42], [43], and [44]. Research in [35] and [36] reviews

TABLE 1. Existing research and survey on artificial intelligence for wireless communication systems.

References	Channel Coding	Reinforcement Learning	Deep Learning	Federated Learning	Classification	Remarks
M. Chen et al., 2020 [35]	x	x	x	✓	x	Reviews and investigated use cases of different federated learning. Suggests solution for wireless factors on federated learning issues.
Y. Liu et al., 2020 [36]	x	x	x	✓	x	Review of federated learning for 6G applications. Highlights the challenges and future direction of federated learning for 6G.
I. F. Akyildiz et al., 2020 [37]	x	✓	✓	x	x	Extensive overview of requirements for 6G and beyond for wireless communication systems. Minor overview of AI in physical layer and wireless systems.
L. I. Khan et al., 2020 [38]	x	✓	x	✓	x	Explores emerging and advancement for 6G wireless communication systems. Suggests machine learning as a key enabler for future technologies in 6G.
J. Kaur et al., 2021 [39]	x	✓	✓	✓	x	Reviews various machine learning techniques at different application layer and the future of machine learning for 6G wireless communication.
K. Arora et al., 2020 [40]	✓	x	x	x	x	A survey of existing channel coding techniques and also channel coding in 5G standards including encoding and decoding methods.
S. Xu, 2020 [41]	✓	x	✓	x	x	In-depth review and survey of state-of-the-art deep learning application and research for channel coding.
A. Ly and Y. -D. Yao, 2021 [42]	✓	x	✓	x	x	Highlights deep learning research on 5G development. Encourages deep learning for 5G and beyond technologies to improve Quality of Service in wireless technologies.
C. Zhang et al., 2020 [43]	✓	x	x	✓	x	Reviews current and emerging applications of artificial intelligence in wireless communication. Highlights artificial intelligence-based techniques for 5G and beyond.
Y. Arjoune and S. Fatuque, 2020 [44]	✓	✓	✓	x	x	Reviews current technologies of AI-based 5G wireless communication techniques. Highlights several advantages and challenges of implementing AI in wireless systems.
Our paper	✓	✓	✓	✓	✓	Review on machine learning techniques with channel coding. Proposes a new classification for machine learning techniques for channel coding.

the techniques in federated learning (FL) and the challenges for future FL implementation in communication systems. The authors in [35] suggest that channel coding can be a suitable solution for several challenges faced by FL.

Articles in [37], [38], and [39] provide insight into the requirements of 5G and beyond for wireless communication systems, including the implementation of ML. The authors in [37] and [39] provide a minor overview of AI, which includes reinforcement learning (RL) and deep learning (DL) for wireless communication systems, whereas the authors in [38] only discuss RL and FL for future developments of 6G.

Article [40] takes a comprehensive look at channel coding techniques, especially for 5G wireless networks. Their comprehensive survey on channel coding techniques does not include any ML techniques for channel coding.

The articles in [41] and [42] review the implementation of DL in channel coding, especially in current and future wireless communication applications. Although other ML techniques, such as RL and FL, are not discussed, they provide a very comprehensive review of DL application and research.

Deep learning implementation for channel coding is thoroughly discussed in [43]. While the authors do not cover other ML techniques, they provide valuable insights into the specific use of DL in wireless communication systems. Finally, The authors of [44] offer a comprehensive analysis of the current and future implementation of both DL and RL for wireless systems. The authors also highlight some of the current research on deep learning for channel coding, providing a broad overview of the current

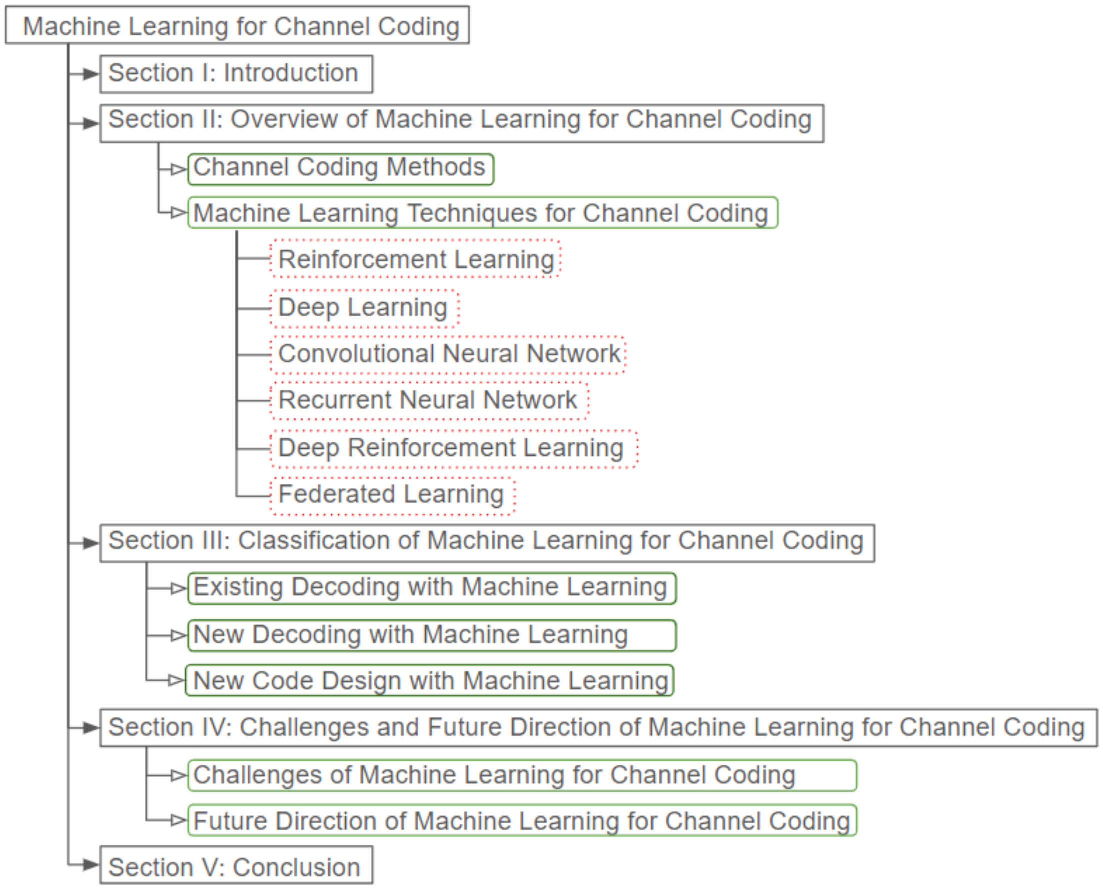
Machine Learning for Channel Coding
Section I: Introduction
Section II: Overview of Machine Learning for Channel Coding
Channel Coding Methods
Machine Learning Techniques for Channel Coding
Reinforcement Learning
Deep Learning
Convolutional Neural Network
Recurrent Neural Network
Deep Reinforcement Learning
Federated Learning
Section III: Classification of Machine Learning for Channel Coding
Existing Decoding with Machine Learning
New Decoding with Machine Learning
New Code Design with Machine Learning
Section IV: Challenges and Future Direction of Machine Learning for Channel Coding
Challenges of Machine Learning for Channel Coding
Future Direction of Machine Learning for Channel Coding
Section V: Conclusion

FIGURE 2. An overview of the content of this paper.

state of ML implementation in wireless communication systems.

In conclusion, the works listed in Table 1 offer a variety of viewpoints on the implementation of ML approaches in wireless communication systems. While some works provide exhaustive overviews of the various ML approaches, others concentrate on certain topics or methodologies. Yet, they contribute to a deeper understanding of the potential benefits and problems of implementing ML in wireless communication systems, with applications to channel coding and other areas in wireless communication systems.

To the best of our knowledge, this is the first study to present a classification for the application of ML for channel coding, as shown in Table 1. Our contributions are as follows:

- We explore current trending ML techniques for channel coding.
- We propose a new type of classification of ML for channel coding purposes and surveyed papers that fall under the class.
- We discussed the current challenges and future trends of ML for channel coding moving forward.

The organisation of this paper is as shown in Fig. 2, which consists of 5 sections. A brief introduction and motivation of ML and channel coding is in Section I. Here, we explore the history of ML or AI in general and the motivation for applying ML for channel coding. Then, Section II provides an overview of ML for channel coding. We start with a review of recent channel coding methods and an overview of ML techniques for channel coding, which we categorised into reinforcement learning, deep learning and federated learning. Our main contribution to this paper is in Section III, where we present a new ML classification for channel coding. Although there are similarities between the reviewed articles on the application of ML in channel coding, there is still no proper terminology or classification for them in the literature. Therefore, we classify ML for channel coding into three types: existing decoding with ML, new decoding with ML and new code design with ML. In section IV, we discuss the challenges and future direction of applying ML in channel coding in terms of the different types of ML methods. Lastly, the conclusion of this paper is drawn in Section V.

II. OVERVIEW OF MACHINE LEARNING FOR CHANNEL CODING

A. CHANNEL CODING METHODS

The purpose of channel coding is to send data or information bits more reliably [58]. Several types of channel coding for wireless networks can be classified into algebraic block

TABLE 2. Summary of some channel coding techniques.

Reference	Methods	Advantage	Disadvantage	Decoding Technique
[45], [46], [47], [48], [49]	Algebraic block codes :- Hamming code, Reed-Muller (RM) code, Bose-Chaudhuri-Hocquenghem codes (BCH codes)	Memoryless and easy to be implemented.	Information cannot be extracted until whole message is received.	Maximum Likelihood Algorithm, Hard-decision, Soft-decision
[46], [50]	Convolutional codes	Improves Bit-Error-Rate (BER) performances	High complexity in decoding and encoding	Viterbi Algorithm
[46], [51]	Turbo codes	Less decoding complexity and achieves near Shannon limit	Poor performance at very low BER. High latency	Iterative soft-decision decoding algorithm
[52], [53]	Fountain Codes :- Luby Transform (LT) codes, Raptor codes	Rateless which allows for large data size for encoding	Not systematic which increases complexity	Belief propagation algorithm
[54], [55]	Polar codes	Simple encoding and decoding algorithm. Easy to implement	Decoding performance is poor due to high latency	Successive Cancellation algorithm, List decoding
[46], [56], [57]	LDPC codes	BER performance approaches Shannon Limit. Low decoding complexity	Requires many mathematical operations per bit	Belief Propagation, Min-sum algorithm

codes, convolutional codes, turbo codes, LDPC codes, fountain codes and polar codes [59]. In the case of 5G standards for channel coding, LDPC codes and polar codes are adopted [40]. There are many types of channel coding techniques as well as their subtypes. Table 2 summarizes some of the more common channel coding techniques used in the history of wireless communication.

Early channel coding techniques, such as the algebraic block codes, focus on error-correcting. It is memoryless and easy to implement; however, the information cannot be extracted until the whole message is received. Following the leap to 3G communication, convolutional codes have become the standard for 3G [60]. Convolutional codes significantly improve BER performances compared to algebraic block codes. However, the major downside of convolutional codes is that the complexity of channel coding is very high.

The introduction of turbo codes realizes the capability of achieving the Shannon limit [61]. The Shannon limit is used to determine the maximum rate of error-free data that can be transmitted theoretically [62]. Knowing this limit will allow us to know the minimum required energy per bit to send information reliably.

The first rateless erasure codes, LT codes, were introduced to offer adaptive and efficient data transmission by generating an unlimited stream of encoded symbols from a finite set of source symbols. Due to the decoder's ability to retrieve the data using the fewest possible encoding symbols, this indicates that LT codes are highly efficient in erasure channels, approaching optimality [53]. Fountain codes or rateless codes are advantageous because they do not fix their code rate before transmission. Therefore, they exhibit universality by being both nearly optimal for any erasure channel and highly efficient as the data length increases [63]. There are some applications of fountain codes that include storage where many files can be transmitted and recovered the same way at every storage device [64].

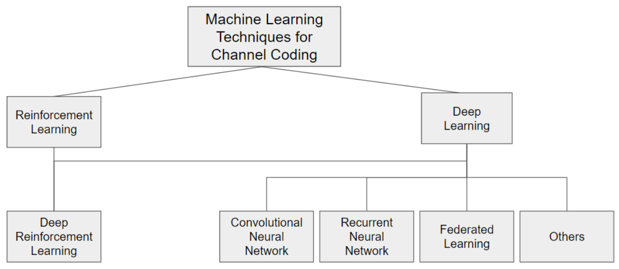
Machine Learning Techniques for Channel Coding
Reinforcement Learning
Deep Learning
Deep Reinforcement Learning
Convolutional Neural Network
Recurrent Neural Network
Federated Learning
Others

FIGURE 3. Machine learning techniques.

Polar codes are a type of codes that has low complexity in encoding and decoding, which was introduced by Arikan [65]. It can achieve symmetry in any binary-input discrete memoryless channel, allowing for the low complexity structure. This also led to the implementation of interleaver in the code design, which has the advantage of improving coding performance, especially in 5G standards [66].

LDPC codes are a type of linear block code that is defined by a parity-check matrix. It can also be represented by a bipartite structure known as Tanner graph or protograph [67]. The main advantage of LDPC codes is that they can get good error performance without using interleavers [68]. QC-LDPC codes are a type of protograph code adopted in 5G wireless communication standards [69]. Less memory is required for QC-LDPC codes as compared to normal LDPC codes, and QC-LDPC codes has the capability of being rateless, but normal LDPC codes does not [70].

B. MACHINE LEARNING TECHNIQUES FOR CHANNEL CODING

The usage of ML to improve channel coding methods is the latest trend. Whether it is used for encoding or decoding or even as an autoencoder, the aid of ML plays a crucial role in overcoming the current limitations of channel coding

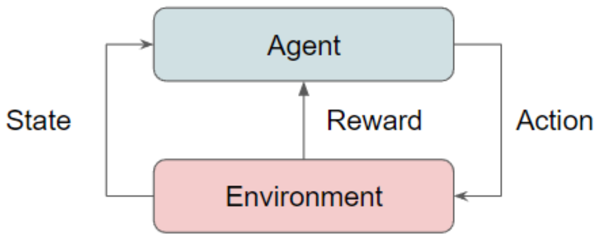

FIGURE 4. MDP representation in a feedback loop between the agent and environment.

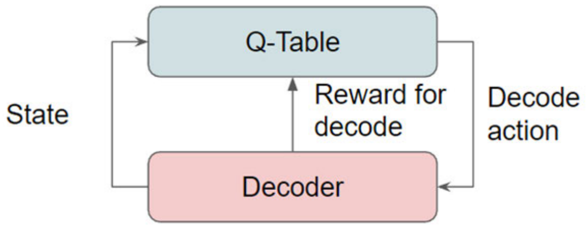

FIGURE 5. Q-learning structure based decoder.

techniques. We categorise ML for channel coding such as shown in Fig. 3. For ML, there are two common techniques under ML, which are RL and DL. Under RL, there is deep reinforcement learning (DRL), which is considered a sub-field of ML that incorporates both RL and DL. Under DL, there is a convolutional neural network (CNN), RNN, FL and others. We define others as any technique that uses DL techniques but is not limited to CNN, RNN and FL techniques in Section III. The theory and application of these ML techniques listed here are further discussed in the following sub-sections.

1) REINFORCEMENT LEARNING

RL is often used to apply learning agents to achieve specific goals in a certain environment [71]. The Markov Decision Process (MDP) is a model that is often used in reinforcement learning. The MDP can be described as an environment where there is an action that an agent can take and be rewarded and its expectation for the action and the next status after the action. This process can be simplified in Fig. 4.

Reward system of an agent can be explained simply by "1" and "−1". When the action taken is desired or successful, it will be rewarded with a positive value such as "1". If it is an undesired action, the reward function will return a negative value such as "−1". The more positive rewards the agent is able to collect will reinforce the action take by the agent, which enhances the policy of the agent.

The agent's decision is often defined by a certain policy. A policy is a function that allows the action to be taken by the agent in the environment. This will in turn affect the reward and state depending on the action taken according to the policy. There are several components to learning a policy in RL. One approach involves using a value-based model that predicts the quality of each state. Another approach uses a direct representation of the policy. Alternatively, a model of the environment can be used, which includes estimated transition and reward functions along with the algorithm. Examples of model-based approaches are lookahead search and trajectory optimization. When there is a model of the environment, the planning process involves interacting with the model to suggest an action. For discrete actions, this often involves generating possible trajectories through a lookahead search. In the case of a continuous action space, trajectory optimization using different controllers can be utilized [72].

The policy gradient method serves as a direct representation of the policy, providing an approach in which the policy itself is optimized directly. Examples of policy gradient methods include stochastic policy gradient, deterministic policy gradient, and actor-critic method. A policy is considered deterministic if the probability of a chosen action is one in all states. A deterministic policy is important because it will determine the effectiveness of the RL algorithm in practical situations as the behaviour is easier to predict [73].

Generally, the state space is defined by S, the action space is A and the state is s.The state s will eventually transition to the update state s'. R is the reward after transition from $s \to s'$ triggered by a certain action, a.

Value-based methods involve constructing a value function that can be used to determine an optimal policy. Examples of value-based methods are Q-learning, deep Q-networks, and multi-step learning. Q-learning or Q value is often known as the optimal sum of rewards associated with the pair of action and state. During the training of reinforcement learning, the Q-learning algorithm denotes the optimal strategy for achieving the best results [74]. The Q-learning algorithm is denoted by

$$Q'(s,a) = Q(s,a) + \alpha \cdot [R + \gamma \cdot \max_{a'} Q(s',a') - Q(s,a)], \tag{1}$$

where α is learning rate, γ is reward discount factor.

Fig. 5 describes the Q-learning algorithm when applied to the MDP environment. This is an example of the usage of the general RL idea in Fig. 4 when applied in channel decoding. Utilization of Q-table is very common in most applications. Q-table is used to store the highest reward value, which can be referred to during the decoding process.

However, a unique Q-learning algorithm proposed in [75] optimized Q-learning for clustering. The clustered Q-learning aims to optimize clustered connecting sets of the Tanner graph of LDPC codes to reduce the learning complexity of decoding.

Advantages of RL include long-term multistep reward optimization. Another reason is that metrics for RL can be easily included. If there is no clear goal for a certain function, an artificial reward can be introduced to aid in RL [76].

2) DEEP LEARNING

DL has wide, various applications in different fields and has also found its way into channel coding applications. It is a

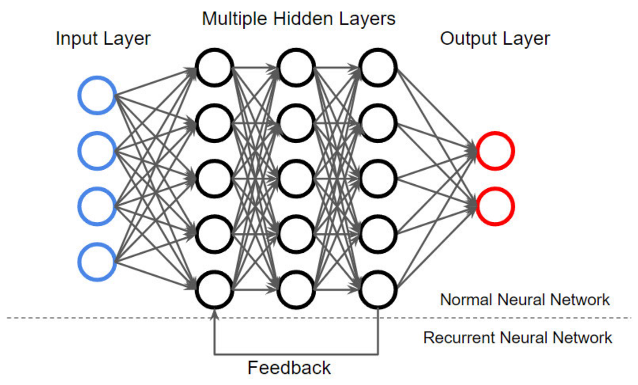

FIGURE 6. An example of deep neural network with 3 hidden layers.[1]

subset of machine learning that employs ANN for learning and is also known as a deep neural network. The DL approach may express extremely nonlinear relationships between input and output vectors by adding more neural processing layers and giving it more training data. DL consists of several neural network types; however, more notable ones for channel coding include CNN and RNN applications.

The neural network may provide a better channel coding solution compared to traditional methods in several ways. These include coding performance, complexity in computing, energy consumption and latency. The model of a simple feed-forward neural network can be expressed as:

$$f_{W_1,W_2,\ldots}(x) = \ldots\varphi_2(W_2\varphi_1(W_1x))\ldots, \tag{2}$$

where f is the model function, x is a model input, and the learnable parameters are denoted as W and scalar-valued nonlinear activation function φ. In the cascade process of the neural network, the linear equation becomes a non-linear equation in shape due to the activation function in the hidden layer. The nonlinear mapping and distributed processing capabilities provide an advantage [77]. A suitable model for channel coding is a feed-forward NN.

Although the distinctive operation of the human brain served as inspiration for the creation of ANNs, these artificial neural networks are not even close to their biological counterparts. ANNs lack the complexity of the brain even though they share two important characteristics with biological neural networks. First, both networks are composed of basic computing units that are closely coupled. Second, the network's functionality is determined by connections between neurons.

ANNs possess the capability to adapt their internal configurations to attain optimal solutions when exposed to sufficient data and appropriate initialization. From the inputs to a system, an ANN is capable of assimilating knowledge from its surroundings, emulating the cognitive processes of the human brain, and subsequently enabling the retrieval of this acquired knowledge at a later stage [78].

[1]Normal deep neural network only refers to the structure above the dotted line. The feedback only applies when mentioning a recurrent neural network structure.

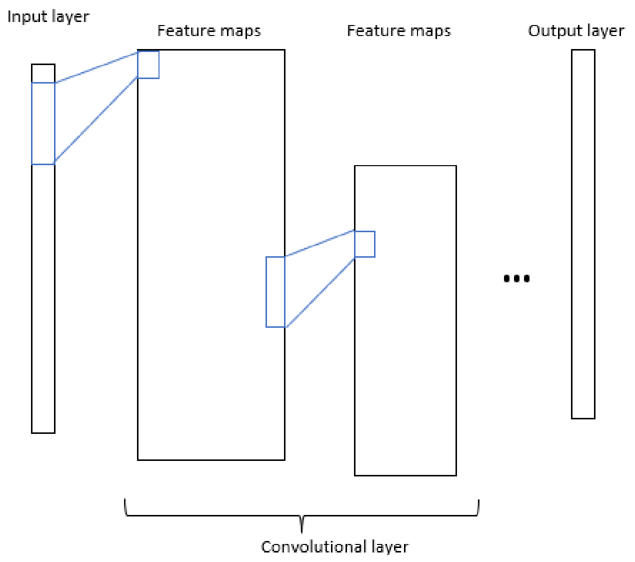

FIGURE 7. A convolutional neural network (CNN) structure.

a: CONVOLUTIONAL NEURAL NETWORK

A feedforward artificial neural network known as a CNN has also been used to analyze visual data effectively. The convolution process dramatically decreases the number of parameters, enabling the network to be deeper with fewer parameters. The hidden layers of CNN are either convolutional or pooling, which simulate the response of a single neuron to a visual stimulus. One of the advantages of CNN is the reduction of downsampling dimensions. This allows the pooling layer to keep useful information while reducing the amount of data [79].

The structure of a CNN is depicted in Fig. 7. In the example of feature extraction, the CNN structure often consists of many convolutional layers or pooling layers known as feature maps. These feature maps obtain a part of the information from the input layer, which is represented in a respective feature map. This process cascades onto the next layer of the feature map depending on the design of the CNN, which is then finally represented as the output layer.

b: RECURRENT NEURAL NETWORK

A recurrent neural network (RNN) is a kind of artificial neural network in which a directed cycle forms between the connections of the neurons. It may display a dynamic temporal behaviour as a result. RNNs, as opposed to feedforward neural networks like CNN, may process arbitrary input sequences using an internal memory. This may thus be used for tasks like voice recognition. The impressive performance of RNN for time series applications may also be applied to a neural network decoder (NND). The structure of RNN is depicted in Fig. 6. There is a feedback signal among the

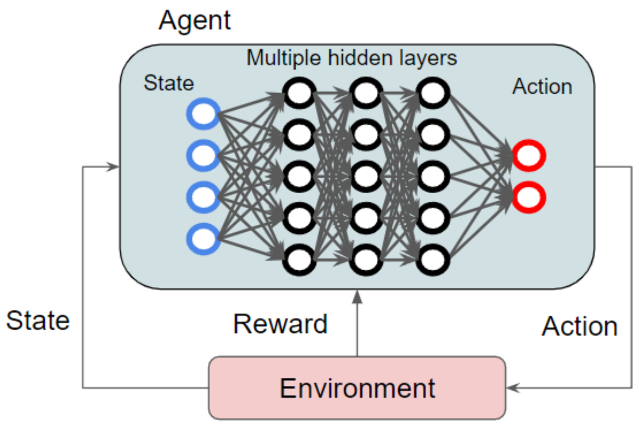

FIGURE 8. Deep reinforcement learning structure.

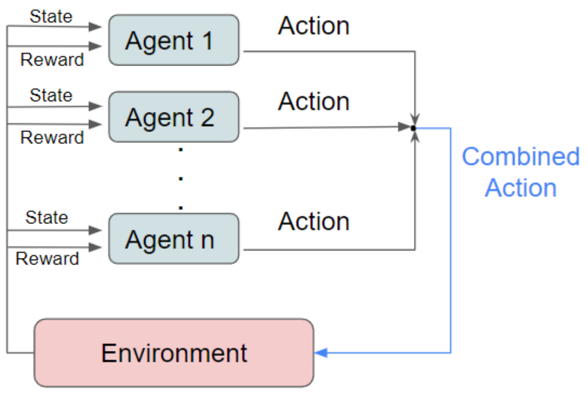

FIGURE 9. Multiple agents interact with the same environment.

hidden layers of the neural network. This is what makes it distinct from normal neural network structures, which are typically feedforward neural network structures.

c: DEEP REINFORCEMENT LEARNING

DRL combines deep neural networks and reinforcement learning. It employs the neural network concept to approximate reinforcement parameters such as the value function. In DRL, a typical structure which dictates the combination of deep learning and reinforcement learning is that it employs a deep Q network to produce a discrete action based on the current state after training through a neural network [80]. The action, in turn, works in the same principle as reinforcement learning, where it will receive a reward for learning from its decision. In essence, the reward updates the state estimate and is updated into the network, as depicted in Fig. 8.

More advanced DRL employs multiple agents interacting with the same environment to get respective state and reward updates [73]. This cooperation between individual agents speeds up the learning process as they interact through the state and reward, as shown in Fig. 9. However, the main challenge in a multi-agent environment is that an agent may affect the behaviour of other agents because they interact with each other. This will change an agent's policy, which may affect the optimized policy of other agents and affect the accuracy of action.

Generalization is the main idea in DRL, which is the ability to obtain good performance in an environment with limited data. The agent needs to perform just like how it performed in a test environment. This is linked to the idea of sample efficiency. Generalization also refers to the ability to achieve good performance in a similar environment but with different dynamics and rewards. This refers to the test environment having certain noise or shifts in features. This is linked to the idea of transfer learning and meta learning [72]. The problem of overfitting arises when there is a limited number of data. This can be solved by increasing the quality of the dataset, allowing the algorithm to rely on the data and reduce asymptotic bias [72].

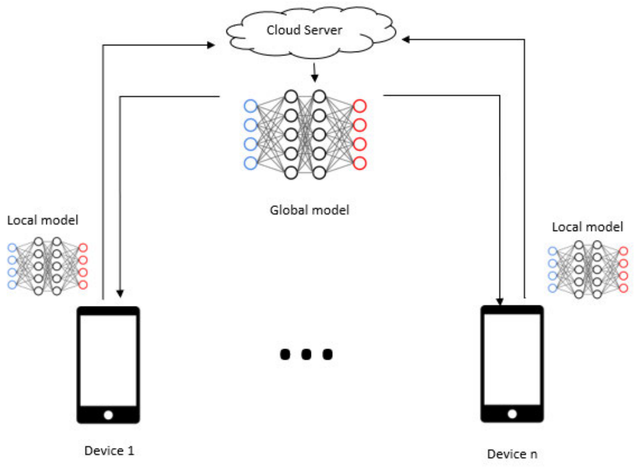

FIGURE 10. A basic federated learning (FL) structure.

C. FEDERATED LEARNING

The concept of FL was proposed by Google, with the idea of building ML models based on data sets distributed across multiple devices while preventing data leakage [81]. FL is ideally suited for data analytics in the IoT and review research addressing privacy concerns [82], bandwidth limitations [83], and power or compute limitations [84].

In FL, mobile devices collaborate to train an ML model that an FL server needs using their local data as shown in Fig. 10. The model changes, i.e. the model's weights, are then sent to the FL server for aggregation. The steps are repeated several times before a desired level of precision is reached. This suggests that FL may be useful for training ML models on mobile edge networks. According to the cloud-centric ML methods, the implementation of FL has the following advantages [85]:

1) Highly efficient use of network bandwidth: For aggregation, participating devices only submit the modified model parameters. It dramatically decreases data communication costs and shortens the supply of network resources.

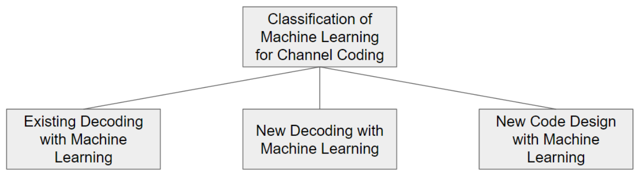
Classification of Machine Learning for Channel Coding
Existing Decoding with Machine Learning
New Decoding with Machine Learning
New Code Design with Machine Learning

FIGURE 11. New classification of machine learning for channel coding.

2) Privacy: provides additional protection for user privacy and has the potential to lessen eavesdropping. Also, more users can participate in collaborative modelling, and thus, more reliable models can be created.
3) Low latency: ML models can be trained and modified on a consistent basis using FL.

In cases of processing and encoding, the server receives a noisy version of the channel inputs transmitted by the users, the transmission mapping and the channel characteristics dictate the relationship between the channel output observed by the server and the encoded model updates [86]. Implementing stochastic quantization and sparsification in FL can reduce the communication load in the model updates. Furthermore, FL models can be designed to boost privacy preservation with respect to the data via, e.g., inducing local noise perturbations relying on the algorithmic foundations of differential privacy [86].

III. CLASSIFICATION OF MACHINE LEARNING FOR CHANNEL CODING

The idea of this classification of ML for channel coding is based on the application of ML for channel coding rather than classification by ML techniques as discussed in the previous section. Fig. 11 shows a new taxonomy that will allow the readers to understand how different ML is applied in channel coding. This is because classifying by ML techniques poses a problem where we are unable to compare the performance of the channel coding itself. For example, reinforcement learning can be used for decoding. Therefore categorizing based on ML can be unfair since the applications are not the same even though they are using the same ML techniques. Fig. 11 depicts a new taxonomy that enables readers to comprehend how various ML algorithms are applied in channel coding.

A. EXISTING DECODING WITH MACHINE LEARNING

Existing decoding with ML can be defined as using existing decoding methods and optimising it with ML techniques such as adjusting weights. This classification also applies to existing channel codes with ML. Generally, a general flow of decoding with ML is shown in Fig. 12. The log-likelihood ratio (LLR) or weights are received by a model which will execute an action, typically the decoding action. The optimization parameters will determine whether to send the updated weight value back to the model to perform a different action. Existing decoding algorithms often is very complex

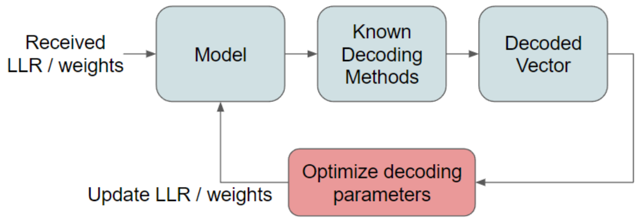

FIGURE 12. General structure for existing decoding with machine learning.

at a low SNR value such as the number states transition increases or in higher order modulation. Other decoding algorithms also faces high complexity problems, such as the Viterbi algorithm for convolutional codes.

A feed-forward deep neural network is proposed by the authors in [87]. The decoder design consists of two structures; a conventional belief propagation (BP) structure and M deep learning blocks. The BP algorithm will update the LLR and send it to the NND. After decoding it at the first NND block, the result will be sent back into the BP decoder, and this process repeats until all M NND blocks are successfully decoded. The main contribution of their proposed method is enabling non-iterative decoding, which has contributed to reducing latency. However, the authors specifically learn to decode polar codes and only investigate short codes. They found that the performance is affected by the degradation through partitioning.

The authors in [88] applied DL to improve the efficiency of the decoding algorithm by identifying and predicting noise samples through noise correlation. Although CNN is often used in applications such as visual imagery and classifications, they consider noise correlation as a feature in their case. Here, CNN performs denoising with relatively low complexity, which makes the decoding process more efficient.

In [88], deep learning is used to improve the efficiency of the decoding algorithm by identifying and predicting noise samples through noise correlation. They applied this in BP decoding for LDPC codes. The noise is modelled so that it can be picked up as output for training. The model is denoted as:

$$\hat{y} = y - \tilde{n} = s + n - \tilde{n} = s + r, \tag{3}$$

where y is the output vector, s is the symbol vector, r is the residual noise and n is the estimated noise.

Although CNN is often used in applications such as visual imagery and classifications, they consider noise correlation as a feature in their case. CNN can perform denoising with relatively low complexity, which will make the decoding process more efficient. In their works, the loss function deployed is,

$$\text{Loss} = \frac{\|r\|^2}{N} + \lambda \left(S^2 + \frac{1}{4}(C-3)^2\right), \tag{4}$$

where r is the residual noise, N is the length of the code block, λ is the scaling factor, S is the skewness, and C is the kurtosis, which is defined as:

$$S = \frac{\frac{1}{N}\sum_{i=1}^{N}(r_i - \bar{r})^3}{(\frac{1}{n}\sum_{i=1}^{N}(r_i - \bar{r})^2)^{\frac{3}{2}}}, \tag{5}$$

$$C = \frac{\frac{1}{N}\sum_{i=1}^{N}(r_i - \bar{r})^4}{(\frac{1}{n}\sum_{i=1}^{N}(r_i - \bar{r})^2)^2}, \tag{6}$$

where r_i is the ith element in the residual noise vector, and $\bar{r}$ is the sample mean.

This loss function was developed for training purposes and incorporates a normality test, which models a Gaussian distribution over the dataset. The authors of [88] showed that the proposed BP-CNN decoder performs better than the standard BP decoding algorithm and is effective at lessening the effects of correlated channel noise. As the CNN acquires knowledge of the noise correlation in order to rectify the errors made by the BP decoder in estimating channel noise, the BP decoder indirectly estimates the channel noise by guessing the coded bits.

The work in [89] proposes three designs of NND, which are multi-layer perceptron (MLP), CNN and RNN. Their NND design for MLP consists of three hidden layers of 64, 32 and 16 nodes, with the length of the encoded binary codeword as the input layer and the length of the information bits as the output layer. As for the NND design for CNN, the CNN model, which is often used in image processing, is modified by changing the input of the layers from 2-D data to 1-D vectors. For the NND design for RNN, RNN is usually used in unsegmented, connected handwriting recognition or speech recognition. They applied the LSTM technique to suit the NND design. The authors found that CNN can achieve better decoding performance against MLP and RNN but at the cost of high computational time. Their work only achieves near-optimal performance when the training data uses 90% of the codewords and low block lengths. Direct use of CNN as decoders is limited by dimensionality and is hard to implement.

In [90], the researchers employ reinforcement learning for bit-flipping decoding for linear block codes that is RM and BCH codes. The researchers discovered that a reinforcement learning method based on fitted Q-learning performed better at bit-flipping decoding. The authors proposed a permutation strategy which sorts the reliability vector and finds the set of vectors which are linearly independent before the decoding process. The authors found that fitted Q-learning provides a better advantage over the standard Q-learning approach because the standard approach uses large memory for the Q-table. It is only useful for short or high rate codes for Q-table.

The authors in [91] exploit the advantage of CNN to train data from a fixed encoding rule and channel probability law so that it can be easily decoded at randomly sampled SNR value. There are many loss functions applied in ML, such as the square loss function, absolute loss function, hinge loss function, and more [92]. We found that a common loss function that is usually applied for channel coding is the square loss function. In [89], the same loss function that is the Mean Squared Error (MSE) based loss function is applied, which is similar to the loss function applied in [91]. However, they model it differently. The loss function employed by [89] is:

$$\text{Loss} = \frac{1}{K}\sum_{i=0}^{K-1}(x_i - \hat{x}_i)^2, \tag{7}$$

where K is the length of information bits, x_i is the target i-th information bit and $\hat{x}_i$ is the estimated neural network output.

However, the loss function employed by [91] is modelled as:

$$\text{Loss} = \frac{1}{\beta\ell}\sum_{i=1}^{\beta}\sum_{j=1}^{\ell}(x_{ij} - \hat{x}_{ij})^2, \tag{8}$$

where β is the number of times for training, ℓ is the set of SNR values, x_{ij} is the message and $\hat{x}_{ij}$ is the neural network output.

From these two loss functions, we can observe that (8) is calculated across β and ℓ while (7) is calculated across K. This is because in [89] the authors are using it for different types of neural network such as MLP, CNN and RNN. They use the same loss function for different types of neural networks in their work.

In [91], the authors train by using valid codewords that are generated in batches. They consider the training from the information bits available regardless of SNR values. It was found that codeword length affects the fitting in deep neural network approaches, limiting learning capability. The computational time of the neural network also becomes a problem if a more complex neural network structure is designed for a longer codeword length, such as increasing layers or neurons.

The advantage of an RNN based decoder is that the weights are tied and set to equal. In [93], RNN was used to optimize decoding relaxation for linear block codes. The RNN decoder strategy was used based on BP methods. The training goal of the RNN is to minimize the multi-loss function. The proposed scheme requires additional memory due to relaxation factors. The RNN-based decoder proposed performs better than conventional BP decoders. However, their study offers a trade-off between implementation complexity and error-correction performance. Another approach with RNN by [94] is to share the decoder weights. Then it uses the quantization of weights for the BP decoding, which can reduce the memory occupied for decoding. A proposed codebook for the quantization of weights in their decoding strategy helps in maintaining high performance. Not only does this achieve less memory overhead, but energy consumption is also reduced. It can be inferred that memory will affect the energy consumption for BP decoding.

A BP-based decoder with RNN is proposed in [93] and [94]. The advantage of an RNN based decoder is that the weights are tied and set to equal. The decoder output is given as [93], The loss function are very similar as they both employ a cross entropy type loss function. The objective is to optimise the weight parameters to produce an N-dimensional output word that approaches the zero codeword as closely as possible. The cross entropy between codeword and output of RNN-BP is given as (9) [94].

$$\text{Loss} = -\frac{1}{N}\sum_{j=1}^{N} u_j \log(o_{j,t}) + (1-u_j)\log(1-o_{j,t}), \tag{9}$$

where N is the N-bit output word, u is the codeword and o is the output in neural network layer at j-th component and time step t.

For RNN, the loss can be computed by adding final marginalization for every time step. Thus, a multiloss variant was suggested in [93] that permits learning of the earliest layers and utilises the gradient update of the backpropagation through time algorithm, and given as (10).

$$\text{Loss} = -\frac{1}{N}\sum_{t=1}^{T}\sum_{j=1}^{N} u_j \log(o_{j,t}) + (1-u_j)\log(1-o_{j,t}), \tag{10}$$

Optimizing decoding with reinforcement learning will reduce the complexity of the algorithm. This is presented by [75] for BP decoding of LDPC codes based on reinforcement learning. Compared to the original Q-learning algorithm in 1, their proposed clustered Q-learning, Q is denoted as:

$$\begin{aligned}\Delta Q(s_u, a_u) = (1-\alpha)Q(s_u, a_u) + \alpha(R(s_u, a_u, s'_u)\\ + \gamma \max_{u', a'_u} Q(s'_u, a'_u)),\end{aligned} \tag{11}$$

where u represents the cluster index s_u and a_u represents the state and action of cluster respectively, s'_u represents the new cluster state. They have shown that clustering the state space efficiently can improve performance and reduce the learning complexity with their proposed clustered Q-learning. The performance of scheduling BP decoders for short-to-moderate length LDPC codes is greatly enhanced by the proposed RL-based decoding framework.

The principle of exponential smoothing is expressed in 11. For channel coding, exponential smoothing is suitable because we want to estimate the average from a sequence of observations [95]. Let W_n be the nth observation of the quantity we a trying to estimate and let $\overline{\mu}^n$ be the estimate of the true mean μ after n observations. A widely used method for computing $\overline{\mu}^{n+1}$ given $\overline{\mu}^n$ and a new observation W^{n+1} is given by:

$$\overline{\mu}^{n+1} = (1-\alpha_n)\overline{\mu}^n + \alpha_n W^{n+1}, \tag{12}$$

where α is the smoothing coefficient.

The authors of [96] also applied CNN in their decoding algorithm. They used successive cancellation (SC) decoding for polar codes. The authors demonstrated that the proposed cascaded CNN-SC decoder overcomes the code length restriction of previous deep learning models that used a single polar decoder. It is discovered that employing the proposed loss function significantly improves the network model's performance by increasing the network model's training accuracy. The loss function deployed in this work is similar to that of [88]. This pattern makes us believe that the loss function modelled by them is suitable for applications of CNN in decoding, although this work applied SC decoding for polar codes whereas in [88] used BP decoding for LDPC codes.

The proposed algorithm in [97] attempts to model their encoder and decoder for federated learning in wireless networks. The model consists of systematic LDPC for error correction at client-side or the transmitter side. The decoder employed on the server-side utilizes a sum-product algorithm to construct a factor-graph to extract the aggregated updates. This is achieved by approximating the maximum probability mass function (PMF) to recover the update and subsequently update the global model. However, this process presents a significant challenge, as the summation of distinct client weights may lead to inaccuracies or errors.

Successive cancellation list (SCL) is also known for its high complexity. Thus, the author in [74] proposed a successive cancellation flip decoding algorithm with Q-learning to overcome the issues, which were to reduce the decoding delay. They also further investigated the decoding delay and found that at high SNR, the decoding delay Q-learning assisted successive cancellation flip decoding is less than the conventional successive cancellation flip decoding, which shows that reinforcement learning succeeded in reducing decoding delay without compromising performance. Decoding attempts of their proposed method is also lower than other algorithms when SNR is low. The frame error rate (FER) of the proposed method is also performing similar to the performance of conventional successive cancellation flip decoding and other decoding algorithms. However, the complexity of the algorithm and the training time is not presented in their paper.

The works in [98] found a new approach for achieving decoder diversity in BP-RNN based decoding schemes for short LDPC codes. The authors further improve the BP-RNN decoding scheme by implementing ordered statistics decoding (OSD) which is a post-processing step that sorts the variable nodes in order to achieve maximum likelihood decoding. The OSD post-processing step is applied when the BP-RNN decoders fail to find a codeword. This will effectively improve the error correction performance. They found that the BP-RNN with OSD post-processing outperforms the traditional BP-RNN decoder. The gain in performance is observed for both different code rates.

The authors in [99] propose a neural network-based approach for path splitting SCL decoder to decode polar codes. They treat the path-splitting selection strategy as a classification problem in which the neural network is able to identify the non-frozen bits accurately. The non-frozen bits are then split, allowing the selection process to be conducted more efficiently. In their findings, they show that the block error rate (BLER) performance are similar to conventional

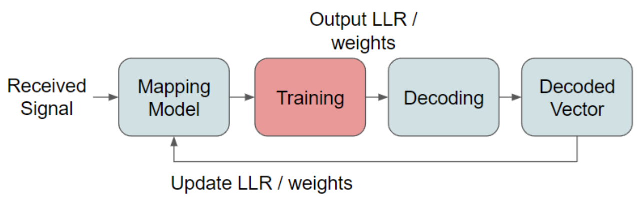

FIGURE 13. General structure for new decoding with machine learning.

SCL decoding methods at different code rates. However, they found that their proposed strategy has lower complexity as compared to other path splitting SCL decoders.

The researchers in [100] propose a BP decoding algorithm that utilizes RL, which incorporates check nodes scheduling policy through Q-learning. It is noted that this is an extension of their previous work in [75]. Compared to their previous work, this work introduces a meta-learning RL scheme for decoding LDPC codes. The meta-training process includes learning a global long-term expected reward or action value function using LLR vectors corresponding to a mixture of SNR values. The LLR vectors corresponding to the SNR values are used to learn a local long-term expected reward or action value function. They found that their proposed algorithm outperforms traditional decoding schemes as it requires fewer check nodes to variable node messages for decoding. A deep Q-learning based scheme of their algorithm was also used for a cluster size of 2, however, the usage of deep Q-learning was not extensively explained in their findings.

B. NEW DECODING WITH MACHINE LEARNING

New decoding with ML can be defined as using ML to design a new decoding method for existing channel coding. The general structure of the new decoding with ML is shown in Fig. 13. The received signals are mapped and trained to recognise the received symbols. The produced LLR or weights will be used, along with ML, for training. Thus, a new form of decoders can be constructed. This new form of decoding with ML may or may not be performing better than current decoding standards, as its performance is often dictated by the loss function considered.

The authors in [101] employ a rectified linear unit (ReLU) activation function in their NND by employing a sigmoid activation function. It is a feed-forward deep neural network. The normalized validation error (NVE) was proposed to measure the NN decoding performance and compare it with MAP decoding, which is given as:

$$\mathrm{NVE}(p_t) = \frac{1}{S}\sum_{s=1}^{S} \frac{\mathrm{BER}_{\mathrm{MAP}}(p_{v,s})}{\mathrm{BER}_{\mathrm{NND}}(p_t, p_{v,s})}, \tag{13}$$

where p_t and p_v are the SNR of the dataset for training and validation, respectively. $\mathrm{BER}_{\mathrm{MAP}}(p_{v,s})$) and $\mathrm{BER}_{\mathrm{NND}}(p_t, p_{v,s})$ will represent the BER through MAP decoding with p_v and BER of NND with the training and validation dataset.

They found that DL-based decoding is very possible as it enables non-iterative decoding. Their proposed method is unable to attain MAP performance if it is not trained on the whole codebook. This affects the NND to detect unseen codewords for random codes. However, this does not affect the performance of unseen codewords of polar codes as it can still be generalized.

A deep learning-based decoder for both Polar and LDPC codes was introduced in [102]. Their designed decoder is able to decode both polar and LDPC codes using the same decoder design. Their proposed unified network architecture contains a special section called the indicator denoted by $\mathbb{I} = \mathrm{LDPC}$, polar with the decoding function,

$$\hat{b} = \begin{cases} \hat{b}_p = f_p(y) = f(y, \mathbb{I} = \text{polar}) \\ \hat{b}_l = f_l(y) = f(y, \mathbb{I} = \text{LDPC}) \end{cases} = f(y, \mathbb{I}), \tag{14}$$

where $\hat{b}$ is the generalized estimation of information bits b, $f_p(y)$ and $f_l(y)$ to be the decoding function of polar codes and LDPC codes, respectively. The loss function chosen is an MSE function, which is used to train the estimated information bits and the original information bits. The loss function can be defined as

$$L = \tfrac{1}{k}(\hat{b} - b)(\hat{b} - b)^T, \tag{15}$$

where k is the number of information bits and $(\cdot)^T$ is the vector transpose operation. They are able to achieve better performance with their proposed method than the conventional BP method in terms of throughput. However, in terms of BER performance, the performance is quite similar.

In [103], an NND for polar codes was also designed. They assumed that the channel state information (CSI) is known at the receiver despite the transmission packet going through fading channels. The received signal, y, at the receiver is denoted as

$$y = h \cdot x + v, \tag{16}$$

where x is the mapped symbols to the fading channels and h is the complex channel coefficient signal, v is the noise component of the model.

The noise component is modelled by a complex Gaussian random variable. Hence, the received signal is further denoted as

$$\hat{x} = \frac{h^*}{|h|^2} \cdot y, \tag{17}$$

where $(\cdot)^*$ is the complex conjugate operator. The log-likelihood ratio, L, is obtained so that it can be decoded by the trained NND to obtain the message. The L is denoted as,

$$\mathrm{L}(\hat{x}) = \ln \tfrac{P(x=0|y)}{P(x=1|y)} = \tfrac{2}{\sigma^2}\mathrm{Re}(y), \tag{18}$$

where $\mathcal{R}e(\cdot)$ is the real part of the complex number $(\cdot)$ and σ^2 is the variance. It is noted that the equation in 18 is only for

BPSK. Hence, the channel L-values can be expressed as:

$$L_{ch}(x|y) = ln\frac{exp[-\frac{y-1^2}{2\sigma^2}]}{exp[-\frac{y+1^2}{2\sigma^2}]} = \frac{2}{\sigma^2} \cdot y \tag{19}$$

The NN model for their proposed system are made up of neuron signals with weighted inputs $w^T \cdot \hat{x}$, which becomes a bias b. The output is $z = w^T \cdot \hat{x} + b$ that is filtered through the ReLU function and sigmoid function. The authors in [103] considered flat fading channels for their proposed NND instead of AWGN channel. However, their findings only discuss transmitting packets and packet error rate as their metric of reliability.

One similarity from all these papers in this section is that their proposed NND employ ReLU function in as their hidden layers and a sigmoid function at the output layer. This is because ReLU function is nonlinear but also very close to linear, which helps for optimization and at the output layer, a sigmoid function will ensure that the output information bits are between zero and one to be interpreted as the probability of transmission of "1".

The researchers in [104] proposed a graph neural network (GNN) decoding scheme for decoding BCH codes and LDPC codes. Previously, the work of the first author proposed polar decoding with DL in [87]. However, this work is quite different from their previous work. Since then, they have proposed GNN decoding scheme is able to outperform traditional BP decoding for BCH codes with fewer iterations. However, in LDPC codes, the proposed GNN scheme achieves similar results to that of traditional BP decoding for AWGN channel. They also pointed out that the proposed decoding scheme has high complexity.

The authors in [105] proposed a lossless turbo source coding scheme that is able to reduce encoding delay compared to conventional turbo source coding. They apply a CNN structure to estimate the optimal compression length that contributes to reducing encoding delay. This is achievable with the use of a classifier that assigns a class to each input message.

In [106], a DNN-based decoding scheme was proposed to decode hamming codes. Compared to conventional hard decision decoding of hamming codes, the proposed decoder was able to achieve better BER performance. Besides BER performance, they also investigated the memory cell requirements for decoding, in which their proposed scheme requires much fewer memory cells. They also investigated the impact of several other factors during the training of DNN-based decoding schemes, such as learning rates, loss function and activation function, to achieve better BER performance. The authors found that the MSE loss function can effectively recognize the difference between the predicted value and true value as compared to the mean absolute error loss function.

C. NEW CODE DESIGN WITH MACHINE LEARNING

New code design with ML is defined as new form of codes designed by ML for encoding and decoding. This is similar to the concept of autoencoders. An autoencoder is a type of neural network that is designed to learn how to replicate its input as accurately as possible in its output. ML approaches can be employed to build new codes instead of simply modifying the encoding and decoding process.

The application of AI in code construction is also very useful as it helps to make a more optimized error correcting code. The works in [107] highlight the potential of DRL for code design, where they model the codes with neural network parameters, θ, before applying RL policies. The DRL for QC-LDPC codes construction were presented in [107]. Their constructed QC-LDPC codes from DRL were compared to normal QC-LDPC codes through the progressive edge growth method and performed similarly under the same parameter decoded by the sum-product algorithm. They applied the policy based on the Monte Carlo tree search (MCTS) method. The MCTS, α_θ is guided by the latest trained neural network f_θ to produce action probabilities, π at each state, s. The returned result from this method is

$$\pi^{(i)}_{\zeta,t} = \alpha_{\theta_{i-1}}(s^{(i)}_{\zeta,t}), \tag{20}$$

for each time step, t in the ζ-th construction process at the i-th iteration. Their proposed method employs MCTS and DNN to steer the code creation process with a long-term perspective, providing flexibility for any code parameters. The authors also suggested that their method has potential for improvement according to their experimental results.

Another advantage of a new code design is that it can be simulated offline [108]. This is presented in [108], where they implemented a code constructor framework through reinforcement learning for different types of codes, that is, linear block codes and polar codes. In linear codes, that is, the BCH codes, their proposed code constructor is able to form a similar performing codes to RM codes and extended BCH codes. For polar codes, the advantage actor critic (A2C), which is a reinforcement learning model, was employed on SCL decoders to produce a good code construction. As compared to genetic algorithm (GA) based code construction, the proposed A2C method has better performance of block error rate. Although they do not consider the training time for code construction, it was proven that code construction is important and is influenced by the decoding performance. Optimizing decoding performance through reinforcement learning will help in constructing a better error correcting code.

In the case of a deep neural network-based autoencoder in [109], they used deep learning to construct an encoder and decoder, an end-to-end autoencoder. The authors found that the average energy constraint is applied to optimize the encoder and decoder model of the autoencoder. Under channel mismatch situations, the proposed autoencoder performs better than the Hamming code, demonstrating the potential of deep learning in the physical layer. The autoencoder's capacity to create an adjustable coding rate enables effective channel matching. Throughout the training of the autoencoder, they implement a flat fading channel consisting of a fading layer and a noise layer in the channel. This is represented as:

$$y = \mathbf{H} \cdot x + w, \tag{21}$$

where x is the output, $\odot$ represents the element-wise multiplication, $\mathbf{H}$ is the input to the fading layer, and w represents the AWGN, and its real and complex multivariate Gaussian distributions is $\mathbb{CN}(0, \delta I)$. They achieved coding gain in the fading channel even with a simple codebook. This is quite similar to the research in [103], especially between (16) and (21) since they both pass through fading channels for their training.

Polar codes construction was done in [110] with the use of deep learning. The authors planned on using the BP decoding method for the polar codes; hence, the training will be deployed with standard stochastic gradient descent (SGD) to find the weights. A binarizer-layer is used so that the code construction $\mathbf{A}$ is a valid formation of conventional polar codes. The probability of a non-frozen bit psoitiion i, $p_i = \delta(A_{soft}, i)$ where A_{soft} is frozen trainable LLR vectors. The code rate with N code length is given as:

$$R_{c,avg} = \frac{1}{N} \sum_{i=1}^{N} \delta(A_{\text{soft}}), \tag{22}$$

Their loss function is a combination of several loss constraints that include low error rates, target rate, convergence behaviour and avoiding minima. The proposed work by the authors in [110] shows significant performance improvements over state-of-the-art building techniques for BP decoding with a limited number of iterations over AWGN and Rayleigh fading channels. In addition to facilitating the learning of new structures, the proposed framework also facilitates traditional system design, such as the choosing of frozen bit-positions of a polar code.

An autoencoder was made in [111] using DRL techniques. The transmitter and receiver of this autoencoder design are implemented with neural network parameters. They proposed an alternating training method where the receiver uses supervised learning; however, the transmitter uses reinforcement learning for learning. The policy deployed here is the Policy Gradient (PG) [112] method. Since their transmitter and receiver already used deep neural network parameters with θ_T and θ_R for weights and bias, the received signal can be estimated using the PG as:

$$\nabla_{\theta_T} J(\mathbf{m}, \mathbf{l}, \mathbf{X_p}) = \frac{1}{S_c} \sum_{i=1}^{S_c} l^{(i)} \nabla_{\theta_T} (\pi_{\theta_T} ({x_p}^{(i)} \mid x^{(i)})), \tag{23}$$

where $J(m, l, X_p)$ is the gradient of the aggregated loss, S_c is the minibatch size, ∇_{θ_T} is the policy, l is the vector of per-example losses $l^{(i)}$, X_p the matrix of pertubed vectors ${x_p}^{(i)}$ and m as message. Although they did not benchmark their autoencoder with any current standard codes, they included noisy conditions for their autoencoder. The researchers in [111] found that in a suitably high, but realistic, training SNR, their proposed feedback system operates as well as a perfect feedback connection. In addition, the researchers proved that their communication method is superior to both quadrature phase-shift keying (QPSK) and a highly tuned higher-order modulation scheme over a radial basis function (RBF) channel.

An RNN was used to construct a channel autoencoder in [113] called Low latency Efficient Adaptive Robust Neural (LEARN) codes. They use a self-supervised method, which optimizes the code, and RNN was applied to gain generalization across different block lengths. Their proposed network architecture is interesting because it features two Gated Recurrent Unit (GRU) at the decoder, which can reduce delay. The output is compiled at the Fully Connected Neural Network (FCNN). Their proposed encoder design consists of a GRU added to an FCNN. It is a dual-layer RNN design with optimal temporary storage capabilities. The structural delay in the decoder, D is the number of bits for look ahead in decoding. The encoder transmits code x_t so that it can decode the message b_t when it receives y_{t+D} at time, t.

Although their proposed method can outperform convolutional codes for short to medium block lengths, it cannot outperform Turbo or LDPC codes unless they introduce an additional mechanism that increases the complexity. It is also limited to a low range of structural delay. An open problem from this autoencoder design was also presented to consider other latency problems, such as computational processing delay.

A federated learning-based autoencoder was also designed for audio semantic information. The proposed autoencoder from [114] combined source coding and channel coding achieves a great convergence compared to conventional coding schemes by 100 times. The training of this autoencoder is based on CNN models. Since the signal is audio semantic information, it is easy to extract features using CNN and deploy them across multiple edge devices. The federated learning training loss function is a normalized root MSE loss function denoted as:

$$\min_{w^j} \sum_{j=1}^{U} \mathcal{L}(w_j, A_j, \hat{A}_j), \tag{24}$$

The parameters encoder and decoder was simultaneously updated so that the loss function can be minimized. Their proposed system can converge effectively without compromising performance. Their proposed autoencoder is trained using FL to increase the accuracy of semantic information extraction and is able to decrease communication overhead significantly. The authors suggested that their research presents a promising approach for training models over audio semantic communication (ASC) architectures in wireless networks with limited resources.

The application of federated learning in channel coding is still somewhat less. However, there is some research that seems to focus on code design with federated learning. In [115], a scheme for model aggregation in federated learning was proposed to reduce model aggregation distortion. Their code design involves both source coding and channel coding. The accuracy of training is higher than that of traditional digital coding. They constructed their own channel coding design where the encoder maps the symbol from a length-k finite field to a length-n lattice. The decoder computes the sum of lattice, which is subsequently mapped back to the finite field. Their proposed method produced a substantially greater channel rate than typical Gaussian MAC capacity. The distortion in the estimator reduces linearly with the number of edge devices. The numerical findings demonstrated that the test accuracy attained by the proposed method is comparable to uncoded transmission and significantly greater than conventional digital coding.

The authors in [116] proposed an FL framework which effectively reduces communication overhead, improves FL performance, and reduces resource consumption. Their proposed FL framework, DipFL, consists of three parts, that is a deep AirComp aggregation (DACA) module, a joint source-channel coding (JSCC) module based on the variational autoencoder (VAE) model, and a personalized mix module. The DACA module enables efficient n-to-1 information aggregation from edge devices to the base station. The JSCC module enables encoding and compression of the transmitted data while reducing the bias of local samples through regularization. The personalized mix module enhances performance by blending local and global models on edge devices. When comparing their findings with the traditional framework, they found that their proposed framework achieved better resource reduction, particularly at a higher number of edge devices. At high SNR, the classification accuracy of their proposed framework is also similar to an error-free transmission method.

The authors in [117] propose an auto-encoder architecture that is interpretable, ensuring scalability to block sizes challenging for ML-based linear block code design approaches. It is a joint design of a linear block code encoder and a gated neural BP decoder. The gated neural BP decoder uses RNN, which employs a low-complexity static decoding strategy. The gated neural BP equations for the message from variable node i to check node j, $\mu_{v_i \to c_j}$ and posteriori LLR, $\widetilde{\lambda_i}$ are:

$$\mu_{v_i \to c_j} = \lambda_i + \sum_{l \neq j} \omega_{i,l}^{(\mu)} \mu_{c_l \to v_i}, \tag{25}$$

$$\widetilde{\lambda_i} = \lambda_i + \sum_{l} \omega_{i,l}^{(\widetilde{\mu})} \mu_{c_l \to v_i}, \tag{26}$$

where the λ_i is the priori LLR, $\omega_{i,l}^{(\mu)}$ and $\omega_{i,l}^{(\widetilde{\mu})}$ are the trainable weights and $\mu_{c_l \to v_i}$ is the messages received by the check nodes j from variable nodes l.

The advantage of this neural belief propagation equation is that the input weights are not required, and the trainable weights do not depend on the destination check nodes. The complexity is reduced compared to the standard neural belief propagation equation. The design of this autoencoder enables linear block codes that are suitable for BP decoding. The proposed autoencoder was found to be agnostic, which means that it does not need any prior knowledge of linear block codes, allowing the construction of codes of arbitrary code and size. The authors evaluated the complexity of different codes and their respective decoders, further illustrating the advantageous performance-to-complexity ratio of the proposed approach.

The works in [118] propose an overhead reduction approach by using RL. They proposed two RL-based schemes for online fountain codes (OFC). The first scheme uses RL to identify the coded symbol degrees from feedback and calls it RL-based degree determination (RL-DD). The second scheme, OFC with no build-up phase using sectioned distribution (OFCNB-SD) is proposed to determine different stages of degree distribution by estimating degree distribution at the beginning of each section. The authors introduce RL-based sectioned distribution (RL-SD) based on OFCNB-SD to solve the overhead minimization problem. Through optimization of the sectioning of the degree distribution, the full recovery overhead is reduced. They compare their work to the estimation-based degree selection scheme and table lookup scheme for OFC. Their findings show that under a limited feedback scenario, they are able to achieve lower full-recovery overhead. The authors provide a detailed analysis of the performance of RL-DD and RL-SD, including the transmission schemes used in simulation, neural network architectures, training parameters, and optimization techniques.

D. SUMMARY

Table 3 shows the summary of the classification of ML for channel coding. This shows that current ML techniques are used in existing decoding with ML (EDML) and new code design with ML (NCML). However, there are very few papers that focus on new decoding with ML (NDML) as compared to that of EDML.

For EDML, RNN is the most popular ML technique for decoding. RNN is often applied in use cases related to time-domain applications. The main concerns of applying RNN involve delay since they are correlated to time. Processing delay means that the latency increase will result in RNN being an undesirable choice for channel coding in terms of decoding performances. However, RNN still remains a popular application in EDML. These works found that the recurrent structure of the RNN will reduce the required parameters to improve the decoding. The RNN structure allows the decoder to use the same weights, thus improving classical BP decoding performance. BP decoding is also very popular and needs to be improved with ML. This may be due to the fact that BP is a commonly used message-passing algorithm for decoding, especially for LDPC codes and polar codes. The combination of improving BP decoding with RNN is also prevalent in EDML. The application of BP decoding with CNN is also applied by treating the decoding of BP as a classification problem. CNN are often used for feature extractions or correlation extractions in applications such as image processing or natural language processing. In cases such as signal, which is often one dimensional, it was found to be able to identify noise and eliminate noise even though they are not directly implemented in the encoder or decoding layer. Such noise can be treated as a feature which can help in the optimization of the decoding process itself.

There are a lot of papers that use deep learning or apply neural networks for channel coding. Deep learning is very popular in NDML and is mostly used to construct new decoding methods, as shown in Table 3. From these papers, we conclude that the reason that deep learning is widely used for NDML is due to the fact that new decoding can be constructed based on the current decoding design. In the cascade process of the neural network, the linear equation becomes a non-linear equation in shape due to the activation function in the hidden layer. Therefore, in the case of an NND, it learns a decoding algorithm which is targeted to decode those trained codes. This is supported in [101], where the authors found that polar codes decoding based NND can decode polar codes well but not random codes, which showed that the generalization of NND is not as good as some other optimized decoding algorithm such as the BP decoding algorithm. However, an NND still performs better for structured codes, such as polar codes, in terms of decoding complexity.

As we classify NCML as a new form of codes designed with ML for channel coding, it is interesting to see how spread out the techniques are. Many different types of ML techniques are applied by many researchers to try and create autoencoders or new codes. Autoencoders are proposed in [109], [111], [113], [114], [117], and [116]. Autoencoders are often proposed as an alternative to existing codes, where the focus is on guiding the proposed autoencoders to design code metrics instead of optimizing the decoding performance. These autoencoders present a potential design for universal codes in the future. New code designs discussed in [107], [108], and [110] however, use ML to design new codes which are based on current existing codes such as polar codes and LDPC codes. These new codes are usually designed to be specifically tailored to current decoding methods, allowing it to achieve better performance. These new codes also inherently overcome some of the potentially flawed designs of standard existing codes.

In the case of supervised learning neural networks, polar codes are very popular to be chosen as the channel coding method [87], [101], [102], [103], [110]. Most of these papers use AWGN as their noisy channel. In papers such as [103], and [101], they identify and predict the loss function to optimize the decoding process. Another method is using the neural network to learn how to map noisy information to its corresponding correct information [87].

There are a lot of RL applications for channel coding. They focus more on optimizing the decoding performance using RL, which contributes to existing decoding in our classification. The common method used in these papers is the Q-learning method. The Q-learning method is popular as it repetitively estimates the observations of the sufficient agents. This allows the solution to be quickly found. According to Table 3, we are able to see the usage of RL even in recent years, it is still feasible to apply RL for decoding.

DRL is the least popular method in terms of ML for channel coding. Current trends of DRL are towards network or resource management purposes instead of channel coding hence there are so few research that applies DRL for channel coding. Another inference that can be made here is that DRL schemes may be too complex for it to be feasible to deploy as an efficient channel coding application. It is interesting to see that the papers found here are both used for new code design with ML.

The application of federated learning is still fairly new; hence, there are not many papers involving the use of federated learning for channel coding. Current trends of federated learning are investigating the convergence time with known or unknown channel state information (CSI). Besides, most federated learning applications also consider the number of clients or edge devices involved during training. However, we only want to look at the current trends of federated learning for channel coding. We are able to see it gaining popularity in recent years as of the writing of this paper, with more papers suggesting applications of FL in channel coding.

IV. CHALLENGES AND FUTURE DIRECTION OF MACHINE LEARNING FOR CHANNEL CODING

This section describes the challenges that are faced by implementing ML for channel coding and also describes the future direction of ML for channel coding. The challenges and future direction are discussed in terms of respective ML techniques, that is reinforcement learning, deep learning and federated learning.

A. CHALLENGES OF MACHINE LEARNING FOR CHANNEL CODING

The outlook of AI toward 6G wireless communication systems is without its challenges–especially the challenges faced by implementing AI in channel coding. Many aspects of channel coding can be improved, such as encoding, decoding, and autoencoders, which may also be improved with the help of AI to make more efficient end-to-end communication channel codes. Wireless communication, especially towards 6G standards, will require faster and higher system capacity, low latency and better reliability. Energy efficiency is also a focus in the future direction of wireless communication. AI plays a crucial role in ensuring these are achieved.

TABLE 3. Summary of classification of machine learning for channel coding.

	Ref.	Channel Coding or Encoder	Decoding Methods	RL	others DL	DRL	CNN	RNN	FL
Existing Decoding with Machine Learning (EDML)	S. Cammerer *et al* (2017) [87]	Polar codes	Partitioned Polar Neural Network Decoding		✓				
	F. Liang *et al* (2018) [88]	LDPC Codes	Belief propagation (BP-CNN) decoders				✓		
	W. Lyu *et al* (2018) [89]	Polar codes	• MLP decoder • CNN decoder • RNN decoder		✓		✓	✓	
	F. Carpi *et al* (2019) [90]	• Reed Muller • BCH codes	Bit flipping decoding with Q learning	✓					
	K. Yashawi *et al* (2019) [91]	• Convolutional Codes • LDPC Codes	• CNN decoder • RNN decoder				✓	✓	
	E. Nachmani *et al* (2019) [93]	• BCH codes • High density Parity Check	• Neural belief propagation (NBP)-RNN • Neural min-sum(NNMS)-RNN • Neural Offset Min Sum (NOMS)-RNN • modified Recursive random (mRRD)-RNN • mRRD-NOMS					✓	
	C. F. Teng (2019) [94]	Polar codes	• DNN - BP • RNN-BP decoder		✓			✓	
	S. Habib *et al* (2020) [75]	LDPC Codes	BP decoders based on Q learning	✓					
	W. Rao *et al* (2020) [96]	Polar codes	CNN-SC decoding				✓		
	T. Huang *et al* (2020) [97]	LDPC Codes	Sum-product algorithm decoder						✓
	X. Wang *et al* (2021) [74]	Polar codes	Successive cancellation flip decoding algorithm with Q-learning	✓					
	J. Rosseel *et al* (2022) [98]	LDPC Codes	RNN based BP with OSD post processing					✓	
	B. Dai *et al* (2023) [99]	Polar Codes	SCL with path selecting strategy		✓				
	S. Habib *et al* (2023) [100]	LDPC Codes	BP Decoding with Q learning	✓		✓			
New Decoding with Machine Learning (NDML)	T. Gruber *et al* (2017) [101]	• Random code • Polar codes	NND		✓				
	Y. Wang *et al* (2018) [102]	• LDPC • Polar codes	Polar- LDPC with DL decoder		✓		✓	✓	
	A. Irawan *et al* (2019) [103]	• Polar Codes	NND		✓				
	S. Cammerer *et al* (2022) [104]	BCH codes LDPC Codes	GNN Decoder		✓				
	S. Manoucheri *et al* (2022) [105]	Turbo Codes	NND				✓		
	Y. Lei *et al* (2023) [106]	Hamming Codes	NND		✓				
New Code Design with Machine Learning (NCML)	M. Zhang *et al* (2018) [107]	QC-LDPC Codes based DRL	Sum-product Algorithm			✓			
	L. Huang *et al* (2019) [108]	• RL based Linear Block Code • GA based Polar codes • A2C based Polar codes	• Box and Match algorithm decoding • SCL-path metrics • SCL-genie	✓					
	J. Xu *et al* (2019) [109]	• Hamming code as benchmark code	Activation function based on ReLU, Linear and SoftMax		✓				
	M. Ebada *et al* (2019) [110]	DL based Polar codes	BP decoding tailored design		✓				
	M. Goutay *et al* (2019) [111]	DRL autoencoder	supervised learning based DNN decoder			✓			
	Y. Jiang *et al* (2020) [113]	LEARN codes	LEARN Decoder- Two GRU with FCNN					✓	
	H. tong *et al* (2021) [114]	ASC autoencoder based CNN	Audio semantic decoder based CNN				✓		✓
	N. Zhang *et al* (2022) [115]	Random quantization code(source code) and nested lattice code (channel code)	Source decoder similar to MMSE estimator and lattice decoder						✓
	G. Laure *et al* (2022) [117]	Linear Block Codes Based Autoencoder	Gated Neural Belief Propagation Decoder					✓	
	D. Chen *et al* (2023) [116]	JSCC based on VAE model	Joint source-channel decoding						✓
	Z. Qin *et al* (2023) [118]	• RL-DD • RL-SD • OFCNB-SD	• OFC decoding • OFCNB decoding	✓					

1) REINFORCEMENT LEARNING

Limitations of reinforcement learning include an overload of states, which may diminish the results. When applying reinforcement learning, resource allocation may become a problem as reinforcement learning is a data-hungry algorithm [119]. With the increase in wireless communication devices and applications, constant updates in a dynamic situation are required, which will result in a lot of resources overloading the states and causing worse results overall. An increase in system uncertainties will cause several optimization difficulties during resource allocation [39]. As the network size and complexity grow, the connection between decisions becomes more intricate. For instance, decisions related to workload balancing involve both discrete (choosing edge servers) and continuous (allocating workload) actions that are interdependent. The complexity of the system intensifies when numerous actions are involved in a single control decision. The relationships between these actions and designing the action space appropriately pose a critical challenge for implementing RL/DRL algorithms [19].

Training and performance evaluation are still among the biggest challenges of reinforcement learning. Typically, a particular model is a simplification of the actual system and could miss the underlying patterns which are created from the simulated data set. Therefore, a more efficient method of producing simulation data is needed to guarantee that the reinforcement learning framework's training and performance assessment align with actual systems [42].

Reinforcement Learning also takes a lot of time to train. Wireless communications depend heavily on the convergence time, and actual networks cannot afford to take too long to learn a successful method [120]. This is because, unlike other ML techniques, RL/DRL algorithms do not involve a distinct training process. The training process occurs through continuous trial-and-error processes until the agent reaches a final state. The agent's performance relies on the historical data, such as executed actions and exploration strategy employed during the learning process [121].

Sparse rewards are also a challenge for reinforcement learning. Due to the sparse distribution of rewards in the environment, it is feasible that an agent may not be paying enough attention to the circumstance to identify reward signals and maximize certain behaviours. This also happens when the environment cannot send out reward signals quickly, so the agent won't act until it is sufficiently near to the objective [122].

The problem of reproducibility arises in DRL, where experimental outcomes are impacted by various hyperparameters such as network architecture, reward scale, and environments. These factors introduce variability and can affect the consistency and replicability of results obtained in DRL applications [123].

2) DEEP LEARNING

Learning to decode more quickly and effectively has been the aim and challenge, especially in Polar Codes. The NND in [101] retrieves transmitted codeword after the learning process, whereas in [124], the complexity is reduced through the neural network. However, this is difficult as the categories for polar codes restrict the exponential structure of codewords for polar codes as compared to that of image recognition. The increasing code length will result in a large number of datasets for training the neural network. In contrast to the 5G and beyond requirements for high-speed and low-latency requirements, deep neural networks often need exorbitant computing [113]. Therefore, there is no suitable method to use DL decoding with the regular 5G standard code length at the moment.

The application of DL decoders and the performance and complexity are still issues in real situations. It is a challenge to identify the trade-off between complexity and performance when combining the conventional method with DL techniques. Hence, with DL, forward error correcting codes can be further investigated in different areas, such as a joint estimator and decoder design, which may provide significant performance gain [125].

A further limitation of DNNs is their inability to perform time-correlated detection and tracking tasks with video streaming. While a DNN must fully link all nodes in hidden layers, the number of parameters often increases exponentially [77]. RNNs depend on the recurrent structure to increase detection performance, although often with a noticeable processing latency.

3) FEDERATED LEARNING

Federated learning differs from traditional centralized deep learning, where the server can access the whole data set. This results in a number of fundamental difficulties that are not present in traditional deep learning [86]. Massive transmissions across rate-limited communication channels are often required for the recurrent exchange of updated models between users and the server, which creates a bottleneck. Besides, federated learning also consists of a huge number of wirelessly connecting power-constraint edge users. The transfer of model updates may be much slower than local computing due to the restricted capacity of wireless networks. In wireless communication, factors such as loss function value, convergence time, energy consumption and reliability can affect federated learning [35]. These challenges are highlighted in this section, and potential solutions will be discussed in the next section.

Loss function value: will affect the bandwidth and computational resources as well as transmit power. When federated learning is applied, this will result in devices to perform federated learning being limited at every iteration and causing errors in local federated learning models due to insufficient power.

Convergence time: The federated learning model will take longer to transmit the parameters under limited energy and transmit power. Energy consumption: This is a significant factor as depending on wireless resources and channel conditions, energy consumption may increase for transmission of local federated learning models.

Reliability: the reliability of wireless is limited by its transmit power, resulting in errors for local federated learning models.

B. FUTURE DIRECTION OF MACHINE LEARNING FOR CHANNEL CODING

Applying ML in channel coding to achieve improvements in classical channel coding opens up several research problems. Generally, the research problem lies in implementing a suitable ML technique for different tasks. It remains an open problem to compare implementation of different ML techniques in channel coding in order to achieve better performance.

1) REINFORCEMENT LEARNING

Moving towards 6G networks, the introduction of massive Multiple-Input Multiple-Output (MIMO) devices has become as coherent as ever. Using beamspace transmission for interference management becomes a challenge to make this possible. Network optimization is often done using DRL, where beamspace parameters and channel information are approximated. Although several solutions target resource allocation, the reduction of complexity for resource allocation using reinforcement learning is required and becomes an excellent direction for research [39].

The challenges of a network environment lacking decision-making information can be overcome with reinforcement learning [126], [127]. For channel coding, reinforcement learning seems to excel in a dynamic environment. Reinforcement learning can reduce the information needed for decision-making, reduce communication overhead, and simplify the encoding and decoding process.

In wireless systems, efficient management of resources, such as power transmission and resource allocation, is crucial to network performance. DRL can overcome situations like CSI estimation since CSI often changes in a dynamic environment [128]. In IoT systems, using the received power from sensor nodes, DRL can allow effective channel estimation [42].

2) DEEP LEARNING

Despite significant research efforts, there are still several difficulties with employing DNNs for decoding due to the exponentially growing computations with code length. Thus, addressing this challenge is a huge leap towards achieving better performance in the 6G standard.

It is evident that deep learning would significantly lower complexity in channel coding without compromising performance. There is a lot of existing research on polar codes with deep learning however, given the capability of LDPC codes, deep learning in LDPC seems to also be a great open research problem to investigate its potential.

CNN cannot yet be utilized directly for channel encoding or decoding but may be used to extract correlated noise characteristics and improve the BP decoding process. Other iterative detection algorithms can be used to improve better decoding performance. In order to accomplish overall energy efficiency savings, deep learning models, such as DNN and CNN, would significantly increase the BER performance, which remains an open research topic to investigate the capability of energy efficiency [129].

Due to the time-correlated environment in which wireless communications are conducted, RNNs have the potential to enhance detection performance but with significant latency. It will thus be preferable to use a low-latency RNN-based decoding technique as a future research direction. On the other hand, another DL study area focuses on integrating decoding and other communication modules. This research area is expected to emerge in the future by using DNN-based approaches in place of conventional, time-consuming methods [41].

3) FEDERATED LEARNING

As discussed in the challenges of federated learning from the previous section, the future direction of federated learning can be steered towards solving or overcoming these challenges. Limited transmit power will result in errors in local federated models, affecting the loss function value and reliability. A possible solution to overcome this is through channel coding and decoding strategies. Channel coding schemes can overcome noise and interference to the transmitted federated learning model parameters, reducing packet error and increasing reliability [130]. An optimized network formation can also improve the loss function value and reliability other than the channel coding approach.

Convergence time is also a major factor in federated learning, which can be limited due to energy, transmit power and bandwidth limitations. This can be improved by improving the model's parameter prediction, channel coding, and decoding. Coding can reduce the transmission delay of the parameters by compressing them into a small number of bits [131].

Energy consumption of wireless devices may vary due to channel conditions or wireless resources such as bandwidth limitations. This will result in more energy being used to transmit model parameters. One suggested solution for this is channel coding. An efficient channel coding method can improve energy efficiency, thus reducing energy consumption during transmission [132]. Besides channel coding, an efficient device and resource scheduling can also overcome energy consumption problems [35].

We can see that several challenges of federated learning can be solved through efficient channel coding schemes. Hence suitable channel coding strategies for an efficient trade-off with federated learning pose an interesting research problem.

V. CONCLUSION

This paper surveyed existing and latest ML techniques for channel coding in wireless communication networks and outlined some of their essential applications. We also proposed a new classification of ML for channel coding applications and provided a perspective on current challenges and future research. We envisioned that 5G-A and 6G wireless systems would be heavily motivated by the advancement and help of ML, particularly with ML for channel coding. To this end, we compiled and highlighted the latest ML applications for channel coding that they believed would become the stepping stones towards enabling 5G-A and 6G. It is expected that the highlighted challenges and future direction will inspire future wireless communications and also encourage readers to look at existing technologies for further improvements.

REFERENCES

[1] X. Lin, "An overview of 5G advanced evolution in 3GPP release 18," 2022, *arXiv:2201.01358*.

[2] I. Rahman, S. M. Razavi, O. Liberg, C. Hoymann, H. Wiemann, C. Tidestav, P. Schliwa-Bertling, P. Persson, and D. Gerstenberger, "5G evolution toward 5G advanced: An overview of 3GPP releases 17 and 18," *Ericsson Technol. Rev.*, vol. 2021, no. 14, pp. 2–12, Oct. 2021.

[3] B. J. Copeland, *Turing: Pioneer of the Information Age*. New York, NY, USA: Oxford, 2012.

[4] A. L. Samuel, "Some studies in machine learning using the game of checkers. II—Recent progress," *IBM J. Res. Develop.*, vol. 11, no. 6, pp. 601–617, Nov. 1967.

[5] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Trans. Inf. Theory*, vols. IT-13, no. 1, pp. 21–27, Jan. 1967.

[6] K. Fukushima and S. Miyake, "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition," in *Competition and Cooperation in Neural Nets*. Cham, Switzerland: Springer, 1982, pp. 267–285.

[7] A. Bovier and V. Gayrard, "Rigorous bounds on the storage capacity of the dilute Hopfield model," *J. Stat. Phys.*, vol. 69, nos. 3–4, pp. 597–627, Nov. 1992.

[8] G. Tesauro, "Temporal difference learning and TD-gammon," *ICGA J.*, vol. 18, no. 2, pp. 1–88, Jun. 1995.

[9] F.-H. Hsu, "IBM's deep blue chess grandmaster chips," *IEEE Micro*, vol. 19, no. 2, pp. 70–81, Jul. 1999.

[10] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, Sep. 1995, doi: 10.1007/BF00994018.

[11] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[12] Y. LeCun. (1998). *The MNIST Database of Handwritten Digits*. [Online]. Available: https://yann.lecun.com/exdb/mnis

[13] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[14] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.

[15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.

[16] F.-Y. Wang, J. J. Zhang, X. Zheng, X. Wang, Y. Yuan, X. Dai, J. Zhang, and L. Yang, "Where does AlphaGo go: From church-turing thesis to AlphaGo thesis and beyond," *IEEE/CAA J. Autom. Sinica*, vol. 3, no. 2, pp. 113–120, Apr. 2016.

[17] N. Kato, Z. Md. Fadlullah, B. Mao, F. Tang, O. Akashi, T. Inoue, and K. Mizutani, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 146–153, Jun. 2017.

[18] H. Mao, M. Alizadeh, I. Menache, and S. Kandula, "Resource management with deep reinforcement learning," in *Proc. 15th ACM Workshop Hot Topics Netw.*, Nov. 2016, pp. 50–56.

[19] D. Zeng, L. Gu, S. Pan, J. Cai, and S. Guo, "Resource management at the network edge: A deep reinforcement learning approach," *IEEE Netw.*, vol. 33, no. 3, pp. 26–33, May 2019.

[20] P. Ahammad, B. Kennedy, P. Ganti, and H. Kolam, "QoE-driven unsupervised image categorization for optimized Web delivery: Short paper," in *Proc. 22nd ACM Int. Conf. Multimedia*, Nov. 2014, pp. 797–800.

[21] M. Usama, J. Qadir, A. Raza, H. Arif, K. A. Yau, Y. Elkhatib, A. Hussain, and A. Al-Fuqaha, "Unsupervised machine learning for networking: Techniques, applications and research challenges," *IEEE Access*, vol. 7, pp. 65579–65615, 2019.

[22] H. Choi, M. Kim, G. Lee, and W. Kim, "Unsupervised learning approach for network intrusion detection system using autoencoders," *J. Supercomput.*, vol. 75, no. 9, pp. 5597–5621, Sep. 2019.

[23] T. Hu and Y. Fei, "QELAR: A machine-learning-based adaptive routing protocol for energy-efficient and lifetime-extended underwater sensor networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 6, pp. 796–809, Jun. 2010.

[24] Y. Lv, Y. Duan, W. Kang, Z. Li, and F.-Y. Wang, "Traffic flow prediction with big data: A deep learning approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 2, pp. 865–873, Apr. 2015.

[25] N. Taherkhani and S. Pierre, "Centralized and localized data congestion control strategy for vehicular ad hoc networks using a machine learning clustering algorithm," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 11, pp. 3275–3285, Nov. 2016.

[26] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

[27] O. W. Samuel, H. Zhou, X. Li, H. Wang, H. Zhang, A. K. Sangaiah, and G. Li, "Pattern recognition of electromyography signals based on novel time domain features for amputees' limb motion classification," *Comput. Electr. Eng.*, vol. 67, pp. 646–655, Apr. 2018.

[28] Ö. T. Demir and E. Björnson, "Channel estimation under hardware impairments: Bayesian methods versus deep learning," in *Proc. 16th Int. Symp. Wireless Commun. Syst.*, Aug. 2019, pp. 193–197.

[29] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cognit. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[30] M. P. C. Fossorier, "Quasi-cyclic low-density parity-check codes from circulant permutation matrices," *IEEE Trans. Inf. Theory*, vol. 50, no. 8, pp. 1788–1793, Aug. 2004.

[31] W. Saad, M. Bennis, and M. Chen, "A vision of 6G wireless systems: Applications, trends, technologies, and open research problems," *IEEE Netw.*, vol. 34, no. 3, pp. 134–142, May 2020.

[32] B. Zong, C. Fan, X. Wang, X. Duan, B. Wang, and J. Wang, "6G technologies: Key drivers, core requirements, system architectures, and enabling technologies," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 18–27, Sep. 2019.

[33] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6G networks: Use cases and technologies," *IEEE Commun. Mag.*, vol. 58, no. 3, pp. 55–61, Mar. 2020.

[34] M. E. Morocho-Cayamcela, H. Lee, and W. Lim, "Machine learning for 5G/B5G mobile and wireless communications: Potential, limitations, and future directions," *IEEE Access*, vol. 7, pp. 137184–137206, 2019.

[35] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Wireless communications for collaborative federated learning," *IEEE Commun. Mag.*, vol. 58, no. 12, pp. 48–54, Dec. 2020.

[36] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang, and D. Niyato, "Federated learning for 6G communications: Challenges, methods, and future directions," *China Commun.*, vol. 17, no. 9, pp. 105–118, Sep. 2020.

[37] I. F. Akyildiz, A. Kak, and S. Nie, "6G and beyond: The future of wireless communications systems," *IEEE Access*, vol. 8, pp. 133995–134030, 2020.

[38] L. U. Khan, I. Yaqoob, M. Imran, Z. Han, and C. S. Hong, "6G wireless systems: A vision, architectural elements, and future directions," *IEEE Access*, vol. 8, pp. 147029–147044, 2020.

[39] J. Kaur, M. A. Khan, M. Iftikhar, M. Imran, and Q. E. U. Haq, "Machine learning techniques for 5G and beyond," *IEEE Access*, vol. 9, pp. 23472–23488, 2021.

[40] K. Arora, J. Singh, and Y. S. Randhawa, "A survey on channel coding techniques for 5G wireless networks," *Telecommun. Syst.*, vol. 73, no. 4, pp. 637–663, Apr. 2020.

[41] S. Xu, *Channel Coding With Deep Learning*. Hoboken, NJ, USA: Wiley, 2020, ch. 1, pp. 265–285.

[42] A. Ly and Y.-D. Yao, "A review of deep learning in 5G research: Channel coding, massive MIMO, multiple access, resource allocation, and network security," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 396–408, 2021.

[43] C. Zhang, Y.-L. Ueng, C. Studer, and A. Burg, "Artificial intelligence for 5G and beyond 5G: Implementations, algorithms, and optimizations," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 10, no. 2, pp. 145–148, Jun. 2020.

[44] Y. Arjoune and S. Faruque, "Artificial intelligence for 5G wireless systems: Opportunities, challenges, and future research direction," in *Proc. 10th Annu. Comput. Commun. Workshop Conf.*, Jan. 2020, pp. 1023–1028.

[45] S. Lin and D. J. Costello, *Error Control Coding*, vol. 2. Upper Saddle River, NJ, USA: Prentice-Hall, 2001.

[46] J. C. Moreira and P. G. Farrell, *Essentials of Error-Control Coding*. Hoboken, NJ, USA: Wiley, 2006.

[47] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error Correcting Codes*, vol. 16. Amsterdam, The Netherlands: Elsevier, 1977.

[48] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 29, no. 2, pp. 147–160, Apr. 1950.

[49] I. S. Reed, *A Class of Multiple-Error-Correcting Codes and the Decoding Scheme*. Lexington, MA, USA: MIT Lincoln Laboratory, 1953.

[50] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Trans. Inf. Theory*, vols. IT-13, no. 2, pp. 260–269, Apr. 1967.

[51] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *Proc. IEEE Int. Conf. Commun.*, Sep. 1993, pp. 1064–1070.

[52] A. Shokrollahi, "Raptor codes," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2551–2567, Jun. 2006.

[53] M. Luby, "LT codes," in *Proc. 43rd Annu. IEEE Symp. Found. Comput. Sci.* Washington, DC, USA: IEEE Computer Society, Feb. 2002, p. 271.

[54] K. Chen, K. Niu, and J. Lin, "Improved successive cancellation decoding of polar codes," *IEEE Trans. Commun.*, vol. 61, no. 8, pp. 3100–3107, Aug. 2013.

[55] S. A. Hashemi, M. Mondelli, S. H. Hassani, C. Condo, R. L. Urbanke, and W. J. Gross, "Decoder partitioning: Towards practical list decoding of polar codes," *IEEE Trans. Commun.*, vol. 66, no. 9, pp. 3749–3759, Sep. 2018.

[56] R. Gallager, "Low density parity-check codes," Ph.D. dissertation, Dept. Elect. Eng., Massachusetts Inst. Tech., Cambridge, MA, USA, 1960.

[57] O. Gazi, *Forward Error Correction via Channel Coding*, 1st ed. Cham, Switzerland: Springer, 2020.

[58] O. Gazi, *Forward Error Correction via Channel Coding*. Cham, Switzerland: Springer, 2020.

[59] T. Venugopal and S. Radhika, "A survey on channel coding in wireless networks," in *Proc. Int. Conf. Commun. Signal Process. (ICCSP)*, Jul. 2020, pp. 0784–0789.

[60] K. Zheng, L. Huang, G. Li, H. Cao, W. Wang, and M. Dohler, "Beyond 3G evolution," *IEEE Veh. Technol. Mag.*, vol. 3, no. 2, pp. 30–36, Aug. 2008.

[61] D. J. Costello and G. D. Forney, "Channel coding: The road to channel capacity," *Proc. IEEE*, vol. 95, no. 6, pp. 1150–1177, Jun. 2007.

[62] C. Shannon, "A mathematical theory of architecture," *Bell Syst. Tech. J.*, vol. 27, pp. 623–656, Jul. 1948.

[63] F. Mehran, K. Nikitopoulos, P. Xiao, and Q. Chen, "Rateless wireless systems: Gains, approaches, and challenges," in *Proc. IEEE China Summit Int. Conf. Signal Inf. Process.*, China, Jul. 2015, pp. 751–755.

[64] D. J. C. MacKay, "Fountain codes," *IEE Proc. Commun.*, vol. 152, no. 6, p. 1062, 2005.

[65] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Trans. Inf. Theory*, vol. 55, no. 7, pp. 3051–3073, Jul. 2009.

[66] V. Bioglio, C. Condo, and I. Land, "Design of polar codes in 5G new radio," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 1, pp. 29–40, 1st Quart., 2021.

[67] J. Thorpe, "Low-density parity-check (LDPC) codes constructed from protographs," *IPN Prog. Rep.*, vol. 42, no. 154, pp. 42–154, Sep. 2003.

[68] 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, Apr. 2016.

[69] S. V. S. Ranganathan, D. Divsalar, and R. D. Wesel, "Quasi-cyclic protograph-based raptor-like LDPC codes for short block-lengths," *IEEE Trans. Inf. Theory*, vol. 65, no. 6, pp. 3758–3777, Jun. 2019.

[70] A. Fitri, K. Anwar, and D. M. Saputri, "Simple rateless codes based on 5G new radio QC-LDPC codes for dynamic networks," in *Proc. IEEE Int. Conf. Signals Syst. (ICSigSys)*, Jul. 2019, pp. 150–155.

[71] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," 1996, *arxiv:9605103*.

[72] V. François-Lavet, P. Henderson, R. Islam, M. G. Bellemare, and J. Pineau, "An introduction to deep reinforcement learning," *Found. Trends Mach. Learn.*, vol. 11, nos. 3–4, pp. 219–354, 2018. [Online]. Available: http://www.nowpublishers.com/article/Details/MAL-071

[73] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.

[74] X. Wang, J. He, J. Li, and L. Shan, "Reinforcement learning for bit-flipping decoding of polar codes," *Entropy*, vol. 23, no. 2, p. 171, Jan. 2021.

[75] S. Habib, A. Beemer, and J. Kliewer, "Learning to decode: Reinforcement learning for decoding of sparse graph-based channel codes," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 33, 2020, pp. 22396–22406.

[76] P. Winder, *Reinforcement Learning*. Sebastopol, CA, USA: O'Reilly Media, 2020.

[77] S. Tomasin, A. Brighente, F. Formaggio, and G. Ruvoletto, "Physical-layer location verification by machine learning," in *Machine Learning for Future Wireless Communications*, F.-L. Luo, Ed. Hoboken, NJ, USA: Wiley, 2020, ch. 20, pp. 425–438.

[78] Y.-S. Park and S. Lek, "Artificial neural networks: Multilayer perceptron for ecological modeling," in *Developments in Environmental Modelling*, vol. 28. Amsterdam, The Netherlands: Elsevier, 2016, pp. 123–140.

[79] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 33, no. 12, pp. 6999–7019, Dec. 2022.

[80] P. Mary, V. Koivunen, and C. Moy, "Reinforcement learning for physical layer communications," 2021, *arXiv:2106.11595*.

[81] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. Artif. Intell. Statist.*, 2017, pp. 1273–1282.

[82] M. Abadi et al., "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2016, pp. 308–318.

[83] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 53rd Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2015, pp. 909–910.

[84] S. Wang, T. Tuor, T. Salonidis, K. K. Leung, C. Makaya, T. He, and K. Chan, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[85] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated learning in mobile edge networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 2031–2063, 3rd Quart., 2020.

[86] T. Gafni, N. Shlezinger, K. Cohen, Y. C. Eldar, and H. V. Poor, "Federated learning: A signal processing perspective," *IEEE Signal Process. Mag.*, vol. 39, no. 3, pp. 14–41, May 2022.

[87] S. Cammerer, T. Gruber, J. Hoydis, and S. T. Brink, "Scaling deep learning-based decoding of polar codes via partitioning," in *Proc. IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–6.

[88] F. Liang, C. Shen, and F. Wu, "An iterative BP-CNN architecture for channel decoding," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 144–159, Feb. 2018.

[89] W. Lyu, Z. Zhang, C. Jiao, K. Qin, and H. Zhang, "Performance evaluation of channel decoding with deep neural networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[90] F. Carpi, C. Häger, M. Martalo, R. Raheli, and H. D. Pfister, "Reinforcement learning for channel coding: Learned bit-flipping decoding," in *Proc. 57th Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2019, pp. 922–929.

[91] K. Yashashwi, D. Anand, S. Raj B Pillai, P. Chaporkar, and K. Ganesh, "MIST: A novel training strategy for low-latency scalable neural net decoders," 2019, *arXiv:1905.08990*.

[92] F. Nie, Z. Hu, and X. Li, "An investigation for loss functions widely used in machine learning," *Commun. Inf. Syst.*, vol. 18, no. 1, pp. 37–52, 2018.

[93] E. Nachmani, E. Marciano, L. Lugosch, W. J. Gross, D. Burshtein, and Y. Be'ery, "Deep learning methods for improved decoding of linear codes," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 119–131, Feb. 2018.

[94] C.-F. Teng, C. D. Wu, A. Kuan-Shiuan Ho, and A. A. Wu, "Low-complexity recurrent neural network-based polar decoder with weight quantization mechanism," in *Proc. ICASSP - IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2019, pp. 1413–1417.
[95] G. Yan and S. Sheng, "An exponential smoothing channel estimation method for MIMO-OFDM system," *J. Phys., Conf. Ser.*, vol. 1871, no. 1, Apr. 2021, Art. no. 012054.
[96] W. Rao, Z. Liu, L. Huang, J. Sun, and L. Dai, "CNN-SC decoder for polar codes under correlated noise channels," in *Proc. IEEE 3rd Int. Conf. Electron. Inf. Commun. Technol.*, Nov. 2020, pp. 748–751.
[97] T. Huang, B. Ye, Z. Qu, B. Tang, L. Xie, and S. Lu, "Physical-layer arithmetic for federated learning in uplink MU-MIMO enabled wireless networks," in *Proc. IEEE Conf. Comput. Commun.*, Jul. 2020, pp. 1221–1230.
[98] J. Rosseel, V. Mannoni, I. Fijalkow, and V. Savin, "Decoding short LDPC codes via BP-RNN diversity and reliability-based post-processing," *IEEE Trans. Commun.*, vol. 70, no. 12, pp. 7830–7842, Dec. 2022.
[99] B. Dai, C. Gao, F. C. M. Lau, and Y. Zou, "Neural network aided path splitting strategy for polar successive cancellation list decoding," *IEEE Trans. Veh. Technol.*, vol. 72, no. 7, pp. 1–5, Jan. 2023.
[100] S. Habib, A. Beemer, and J. Kliewer, "RELDEC: Reinforcement learning-based decoding of moderate length LDPC codes," *IEEE Trans. Commun.*, vol. 71, no. 10, pp. 5661–5674, Oct. 2023.
[101] T. Gruber, S. Cammerer, J. Hoydis, and S. ten Brink, "On deep learning-based channel decoding," 2017, *arXiv:1701.07738*.
[102] Y. Wang, Z. Zhang, S. Zhang, S. Cao, and S. Xu, "A unified deep learning based polar-LDPC decoder for 5G communication systems," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2018, pp. 1–6.
[103] A. Irawan, G. Witjaksono, and W. K. Wibowo, "Deep learning for polar codes over flat fading channels," in *Proc. Int. Conf. Artif. Intell. Inf. Commun. (ICAIIC)*, Feb. 2019, pp. 488–491.
[104] S. Cammerer, J. Hoydis, F. A. Aoudia, and A. Keller, "Graph neural networks for channel decoding," in *Proc. IEEE Globecom Workshops*, Dec. 2022, pp. 486–491.
[105] S. Manouchehri, J. Haghighat, M. Eslami, and W. Hamouda, "A delay-efficient deep learning approach for lossless turbo source coding," *IEEE Trans. Veh. Technol.*, vol. 71, no. 6, pp. 6704–6709, Jun. 2022.
[106] Y. Lei, M. He, H. Song, X. Teng, Z. Hu, P. Pan, and H. Wang, "A deep-neural-network-based decoding scheme in wireless communication systems," *Electronics*, vol. 12, no. 13, p. 2973, Jul. 2023.
[107] M. Zhang, Q. Huang, S. Wang, and Z. Wang, "Construction of LDPC codes based on deep reinforcement learning," in *Proc. 10th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2018, pp. 1–4.
[108] L. Huang, H. Zhang, R. Li, Y. Ge, and J. Wang, "AI coding: Learning to construct error correction codes," *IEEE Trans. Commun.*, vol. 68, no. 1, pp. 26–39, Jan. 2020.
[109] J. Xu, W. Chen, B. Ai, R. He, Y. Li, J. Wang, T. Juhana, and A. Kurniawan, "Performance evaluation of autoencoder for coding and modulation in wireless communications," in *Proc. 11th Int. Conf. Wireless Commun. Signal Process. (WCSP)*, Oct. 2019, pp. 1–6.
[110] M. Ebada, S. Cammerer, A. Elkelesh, and S. ten Brink, "Deep learning-based polar code design," in *Proc. 57th Annu. Allerton Conf. Commun., Control, Comput.*, Sep. 2019, pp. 177–183.
[111] M. Goutay, F. A. Aoudia, and J. Hoydis, "Deep reinforcement learning autoencoder with noisy feedback," in *Proc. Int. Symp. Model. Optim. Mobile, Ad Hoc, Wireless Netw. (WiOPT)*, Jun. 2019, pp. 1–6.
[112] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[113] Y. Jiang, H. Kim, H. Asnani, S. Kannan, S. Oh, and P. Viswanath, "LEARN codes: Inventing low-latency codes via recurrent neural networks," *IEEE J. Sel. Areas Inf. Theory*, vol. 1, no. 1, pp. 207–216, May 2020.
[114] H. Tong, Z. Yang, S. Wang, Y. Hu, W. Saad, and C. Yin, "Federated learning based audio semantic communication over wireless networks," in *Proc. IEEE Global Commun. Conf.*, Dec. 2021, pp. 1–6.
[115] N. Zhang, M. Tao, J. Wang, and S. Shao, "Coded over-the-air computation for model aggregation in federated learning," *IEEE Commun. Lett.*, vol. 27, no. 1, pp. 160–164, Jan. 2023.
[116] D. Chen, M. Lei, M.-M. Zhao, A. Liu, and S. Sheng, "Deep learning based coded over-the-air computation for personalized federated learning," in *Proc. IEEE 98th Veh. Technol. Conf.*, Oct. 2023, pp. 1–5.
[117] G. Larue, L.-A. Dufrene, Q. Lampin, H. Ghauch, and G. R. Othman, "Neural belief propagation auto-encoder for linear block code design," *IEEE Trans. Commun.*, vol. 70, no. 11, pp. 7250–7264, Nov. 2022.
[118] Z. Qin, Z. Fei, J. Huang, Y. Wang, M. Xiao, and J. Yuan, "Reinforcement-Learning-Based overhead reduction for online fountain codes with limited feedback," *IEEE Trans. Commun.*, vol. 71, no. 7, pp. 3977–3991, Jul. 2023.
[119] K. Arulkumaran, M. Peter Deisenroth, M. Brundage, and A. Anthony Bharath, "A brief survey of deep reinforcement learning," 2017, *arXiv:1708.05866*.
[120] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, "Optimal and scalable caching for 5G using reinforcement learning of space-time popularities," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 180–190, Feb. 2018.
[121] M. S. Frikha, S. M. Gammar, A. Lahmadi, and L. Andrey, "Reinforcement and deep reinforcement learning for wireless Internet of Things: A survey," *Comput. Commun.*, vol. 178, pp. 98–113, Oct. 2021.
[122] L. Lei, Y. Tan, K. Zheng, S. Liu, K. Zhang, and X. Shen, "Deep reinforcement learning for autonomous Internet of Things: Model, applications and challenges," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 3, pp. 1722–1760, 3rd Quart., 2020.
[123] Y. Li, "Reinforcement learning in practice: Opportunities and challenges," 2022, *arXiv:2202.11296*.
[124] W. J. Gross, N. Doan, E. N. Mambou, and S. A. Hashemi, "Deep learning techniques for decoding polar codes," in *Machine Learning for Future Wireless Communications*, F.-L. Luo, Ed. Hoboken, NJ, USA: Wiley, 2020, ch. 15, pp. 287–301.
[125] X. You et al., "Towards 6G wireless communication networks: Vision, enabling technologies, and new paradigm shifts," *Sci. China Inf. Sci.*, vol. 64, no. 1, pp. 1–74, Jan. 2021.
[126] S. Chen, J. Chen, and J. Chen, "A deep reinforcement learning based network management system in smart identifier network," in *Proc. 4th Int. Conf. Digit. Signal Process.*, Jun. 2020, pp. 268–273.
[127] M. Abbasi, A. Shahraki, M. Jalil Piran, and A. Taherkordi, "Deep reinforcement learning for QoS provisioning at the MAC layer: A survey," *Eng. Appl. Artif. Intell.*, vol. 102, Jun. 2021, Art. no. 104234.
[128] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3133–3174, 4th Quart., 2019.
[129] F.-L. Luo, *Machine Learning for Future Wireless Communications*. Hoboken, NJ, USA: Wiley, 2020.
[130] L. U. Khan, W. Saad, Z. Han, E. Hossain, and C. S. Hong, "Federated learning for Internet of Things: Recent advances, taxonomy, and open challenges," *IEEE Commun. Surveys Tuts.*, vol. 23, no. 3, pp. 1759–1799, 3rd Quart., 2021.
[131] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, "Federated learning with quantization constraints," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process. (ICASSP)*, May 2020, pp. 8851–8855.
[132] P. Vigneswari and S. Sivakumari, "Performance analysis of iterative minsum message passing decoding algorithm for 5G NR LDPC codes," in *Proc. 3rd Int. Conf. Intell. Commun. Technol. Virtual Mobile Netw. (ICICV)*, Feb. 2021, pp. 142–146.

HEIMRIH LIM MENG KEE received the bachelor's degree in electronics systems engineering from Universiti Teknologi Malaysia (UTM), in 2021, where he is currently pursuing the Ph.D. degree in communications engineering. He has worked on projects related to robotics and artificial intelligence (AI). His current research interests include AI, robotics, image processing, and wireless communication.

NORULHUSNA AHMAD (Senior Member, IEEE) received the master's degree in electrical engineering (telecommunication) and the Ph.D. degree in electrical engineering from Universiti Teknologi Malaysia (UTM), in 2003 and 2014, respectively. She is currently a Senior Lecturer with the Faculty of Artificial Intelligence, UTM Kuala Lumpur. She has experience as a researcher for more than 15 years. She has published international journal articles/conference proceedings in the areas of wireless communication systems, multiple access systems, machine learning, cognitive radio, the Internet of Things (IoT), and rural communication. Her current research interests include future wireless communication systems, massive IoT technologies, UAV communication, deep learning applications, multiple access techniques, and image processing.

MOHD AZRI MOHD IZHAR (Senior Member, IEEE) received the M.Eng. degree in electrical engineering (communications) from The University of Sheffield, U.K., in 2008, and the Ph.D. degree in electrical engineering from Universiti Teknologi Malaysia (UTM), Malaysia, in 2014.

Since 2014, he has been a Senior Lecturer with the UTM Kuala Lumpur Campus. On leave from UTM, he visited the Southampton Wireless Group, University of Southampton, U.K., for two years, in 2015, where he was appointed as a Research Fellow for 1.5 years with the Centre for Vision, Speech and Signal Processing (CVSSP), in 2019. His current research interests include channel coding, coding theory, joint source-channel coding, coded modulation, cooperative communications, quantum communications, and 3D audio processing. He received the Mappin Medal Award for outstanding academic performance and the Institute of Electrical and Electronics Engineers (IEEE) Prize for the best communications-related final-year project from The University of Sheffield.

KHOIRUL ANWAR (Senior Member, IEEE) received the B.Eng. degree in electrical engineering (telecommunications) from Institut Teknologi Bandung (ITB), Indonesia, in 2000, and the M.Eng. and Ph.D. degrees from Nara Institute of Science and Technology (NAIST), Japan, in 2005 and 2008, respectively.

From 2008 to 2016, he was an Assistant Professor with the Information Theory and Coding Theory Laboratory, Japan Advanced Institute of Science and Technology (JAIST), Japan. Since 2016, he has been the Director of the University Center of Excellence for Advanced Intelligent Communications (AICOMS), Telkom University, Indonesia, and Head of Beyond 5.5G Laboratory (B5.5G Lab). He is also the Vice Chairperson of Asia Pacific Telecommunity Wireless Group (AWG), from 2019 to 2025. His research interests include the development of classical and quantum information theory and coding theory for 5G, 5G-advanced, and 6G, including the quantum key distribution (QKD). He was a recipient of the Best Student Paper Award from the IEEE Radio and Wireless Symposium (RWS'06), USA, in 2006; the Best Paper Award from Indonesian Student Association (ISA 2007), Kyoto, Japan, in 2007; the Best Paper Presenter Award for the Advanced Technology in International Conference on Sustainability for Human Security (SUSTAIN), Kyoto, in October 2011; the Indonesian Diaspora "Award for Innovation," Congress of Indonesian Diaspora, USA, in July 2012; the Achmad Bakrie Award 2014, Jakarta, in December 2014; the Anugerah of Internationally Recognized Contributions from the Governor of West Java, Indonesia, in December 2016; the National Achievement Award by UKP-PIP Pancasila, Jakarta, in August 2017; the Best Paper Award from IEEE TAFGEN 2018, Malaysia, in July 2018; the TAYTB Award 2019 of Indonesia's Award Winning Inventor, in August 2019; and consecutive Best Lecturer Awards from Telkom University, in August 2019, August 2020, September 2021, and September 2022.

SOON XIN NG (MICHAEL) (Senior Member, IEEE) received the B.Eng. degree (Hons.) in electronic engineering and the Ph.D. degree in telecommunications from the University of Southampton, U.K., in 1999 and 2002, respectively.

From 2003 to 2006, he was a Postdoctoral Research Fellow working on collaborative European research projects known as SCOUT, NEWCOM, and PHOENIX. Since August 2006, he has been a member of the academic staff with the School of Electronics and Computer Science, University of Southampton. He was the Programme Leader of electrical and electronic engineering (EEE), from 2018 to 2021. He has been the ECS Doctoral Programme Director with the University of Southampton, since 2021. He is currently a Professor of Next Generation Communications at the University of Southampton. He was involved in the OPTIMIX and CONCERTO European Projects and the IU-ATC and UC4G Projects. He was the Principal Investigator of an EPSRC Project on Cooperative Classical and Quantum Communications Systems. His research interests include adaptive coded modulation, coded modulation, channel coding, space-time coding, joint source and channel coding, iterative detection, OFDM, MIMO, cooperative communications, distributed coding, quantum communications, quantum error correction codes, joint wireless-and-optical-fiber communications, game theory, artificial intelligence, and machine learning. He has published more than 290 articles and coauthored two John Wiley/IEEE Press books in this field.

Dr. Ng is a fellow of the Higher Education Academy, U.K., and a Chartered Engineer and a fellow of the IET. He is one of the founders and the officer of the IEEE Quantum Communications and Information Technology Emerging Technical Subcommittee (QCIT-ETC). He was the IEEE ComSoc Representative at the IEEE Nanotechnology Council (NTC), from 2020 to 2021. He acted as the TPC/track/workshop chairs for various conferences. He is an Editor of *Quantum Engineering*. He was a Guest Editor for the Special Issues of IEEE Journal on Selected Areas in Communication and an Editor of IEEE Access and the *KSII Transactions on Internet and Information Systems*.

• • •